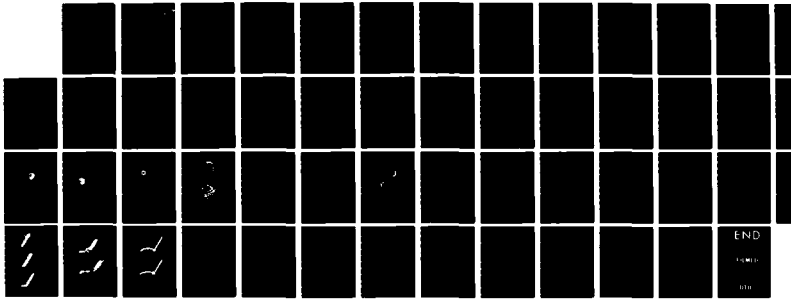
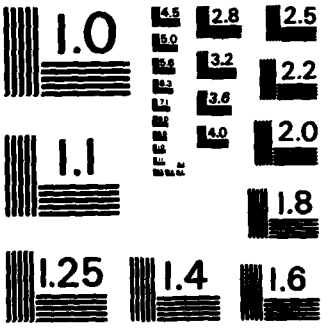
AD-A160 204 A TWO-DIMENSIONAL MESH MOVING TECHNIQUE FOR TIME DEPENDENT PARTIAL DIFFER..(U) RENSSELAER POLYTECHNIC INST TROY NY DEPT OF MATHEMATICAL SCIE.. 1/1

UNCLASSIFIED D C ARNEY ET AL. APR 85 AFOSR-TR-85-0826 F/G 12/1 NL





MICROCOPY RESOLUTION TEST CHART
NATIONAL BUREAU OF STANDARDS - 1963 - A

Unclass

SECURITY CL

AD-A160 204

UMENTATION PAGE

DTIC ELECTE OCT 15 1985 S A

1a. REPORT S	1b. RESTRICTIVE MARKINGS
2a. SECURITY CLASSIFICATION AUTHORITY: UNCLASSIFIED	3. DISTRIBUTION/AVAILABILITY OF REPORT: Approved for public release; distribution unlimited.
2b. DECLASSIFICATION/DOWNGRADING SCHEDULE	
4. PERFORMING ORGANIZATION REPORT NUMBER(S): N/A	5. MONITORING ORGANIZATION REPORT NUMBER(S): AFOSR-TR- 85-0826
6a. NAME OF PERFORMING ORGANIZATION: Rensselaer Polytechnic Institute; 6b. OFFICE SYMBOL (If applicable):	7a. NAME OF MONITORING ORGANIZATION: AFOSR
6c. ADDRESS (City, State and ZIP Code): Department of Mathematical Sciences, Troy, NY 12181	7b. ADDRESS (City, State and ZIP Code): Bolling AFB, D.C. 20332-6448
8a. NAME OF FUNDING/SPONSORING ORGANIZATION: AFOSR; 8b. OFFICE SYMBOL (If applicable): NM	9. PROCUREMENT INSTRUMENT IDENTIFICATION NUMBER: AFOSR-80-0192
8c. ADDRESS (City, State and ZIP Code): Bldg. 410, Bolling AFB, D.C. 20332-6448	10. SOURCE OF FUNDING NOS.

PROGRAM ELEMENT NO.	PROJECT NO.	TASK NO.	WORK UNIT NO.
61102F	2304	A3	

11. TITLE (Include Security Classification): A two-dimensional mesh moving technique for time dependent partial differential equations

12. PERSONAL AUTHOR(S): D. C. Arney and J. E. Flaherty

13a. TYPE OF REPORT	13b. TIME COVERED	14. DATE OF REPORT (Yr., Mo., Day)	15. PAGE COUNT
Interim	FROM _____ TO _____	April 1985	11

16. SUPPLEMENTARY NOTATION: This document

17. COSATI CODES		
FIELD	GROUP	SUB. GR.
	XXXXXXXXXX	

18. SUBJECT TERMS (Continue on reverse if necessary and identify by block number): Mesh moving, finite difference scheme, error indication, error clustering, hyperbolic partial differential equations

19. ABSTRACT (Continue on reverse if necessary and identify by block number)

We discuss an adaptive mesh moving technique that can be used with a finite difference or finite element scheme to solve initial-boundary value problems for vector systems of partial differential equations in two space dimensions. The mesh moving technique is based on an algebraic node movement function determined from the geometry and propagations of regions having significant discretization error indicators. Our procedure is designed to be flexible, so that it can be used with many existing finite difference and finite element methods. To test the mesh moving algorithm, we implemented it in a system code with an initial mesh generator and a MacCormack finite difference scheme on quadrilateral cells for hyperbolic vector systems of conservation laws. Results are presented for several computational examples. The moving mesh scheme reduces dispersive errors near shocks and wave fronts and thereby reduces the grid requirements necessary to compute accurate solutions while increasing computational efficeincy. (Additional keywords)

DTIC FILE COPY

20. DISTRIBUTION/AVAILABILITY OF ABSTRACT	21. ABSTRACT SECURITY CLASSIFICATION
UNCLASSIFIED/UNLIMITED ☒ SAME AS RPT. ☒ DTIC USERS ☐	Unclassified

22a. NAME OF RESPONSIBLE INDIVIDUAL	22b. TELEPHONE NUMBER (Include Area Code)	22c. OFFICE SYMBOL
Dr. Marc Q. Jacobs	(202)767-4940	NM

DD FORM 1473, 83 APR — EDITION OF 1 JAN 73 IS OBSOLETE.

Unclassified

SECURITY CLASSIFICATION OF THIS PAGE

A TWO-DIMENSIONAL MESH MOVING TECHNIQUE FOR TIME DEPENDENT PARTIAL DIFFERENTIAL EQUATIONS[1]

David C. Arney
Department of Mathematics
United States Military Academy
West Point, NY 10996
and
Department of Mathematical Sciences
Rensselaer Polytechnic Institute
Troy, NY 12180-3590

and

Joseph E. Flaherty
Department of Computer Science
Rensselaer Polytechnic Institute
Troy, NY 12180-3590

AMS Subject Classification: 65P05, 76J99

Number of pages: 51

Number of figures: 20

Number of tables: 0

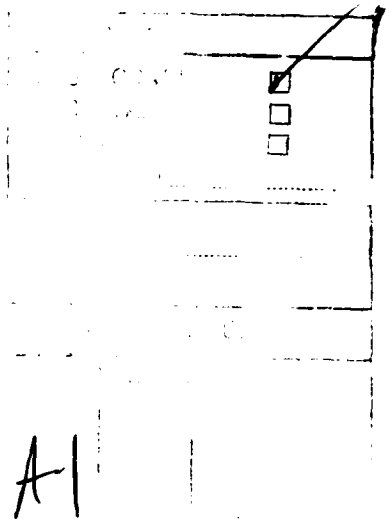
Key Words: mesh moving, finite difference scheme, error indication, error clustering, hyperbolic partial differential equations

[1]The authors were partially supported by the U. S. Army Research Office under Contract Number DAAG29-82-K-0197 and the U. S. Air Force Office of Scientific Research, Air Force Systems Command, USAF, under Grant Number AFOSR 80-0192.

Running title: 2-D Mesh Moving Technique

Send Proofs to:

Professor Joseph E. Flaherty
Computer Science Department
Rensselaer Polytechnic Institute
Troy, NY 12181



A-1

ABSTRACT

We discuss an adaptive mesh moving technique that can be used with a finite difference or finite element scheme to solve initial-boundary value problems for vector systems of partial differential equations in two space dimensions and time. The mesh moving technique is based on an algebraic node movement function determined from the geometry and propagation of regions having significant discretization error indicators. Our procedure is designed to be flexible, so that it can be used with many existing finite difference and finite element methods. To test the mesh moving algorithm, we implemented it in a system code with an initial mesh generator and a MacCormack finite difference scheme on quadrilateral cells for hyperbolic vector systems of conservation laws. Results are presented for several computational examples. The moving mesh scheme reduces dispersive errors near shocks and wave fronts and thereby reduces the grid requirements necessary to compute accurate solutions while increasing computational efficiency.

1. INTRODUCTION

Mesh moving is an adaptive technique that has been used successfully to improve the accuracy of both finite element and finite difference schemes for a variety of time dependent problems in one (cf., e.g., [1,2,3,12,13,15,19,22,25,31]) and two (cf., e.g., [10,30,31,33]) space dimensions. The essential idea is to move the mesh either to minimize some quantity, such as the discretization error, or to follow some local nonuniformity, such as a wave front. This generally reduces dispersive errors and Courant number restrictions.

In one dimension Hyman [25] described a mesh moving scheme that minimized the time variation of the solution at the nodes. This scheme used finite difference approximations for solving hyperbolic conservation laws. Davis and Flaherty [12] and Adjerid and Flaherty [1] developed finite element codes for parabolic systems that moved a mesh so as to equidistribute the spatial component of the discretization error. Miller et al. [19,27,28] simultaneously determined the numerical solution and the node positions using a finite element method that minimized the residual for parabolic problems. Bell and Shubin [3] solved the Euler-Lagrange equations of an extremizing functional and used a finite difference scheme to solve hyperbolic conservation laws. All of these schemes have successfully demonstrated that mesh moving can reduce discretization error and provide improvements in computational efficiency for one-dimensional problems.

With some modification the methods of Adjerid and Flaherty [1], Hyman [25], and Miller et al. [19,27,28], can be extended to higher

dimensions; however, many other mesh moving techniques are not directly applicable to two- and three-dimensional problems. One difficulty is that equidistribution strategies fail to produce unique solutions. Brackbill and Saltzman [10,33] have overcome this problem by adding the constraints of mesh smoothness and orthogonality to a variational problem.

A successful mesh moving scheme for higher dimensional problems that is somewhat similar to the method presented here is the algorithm of Rai and Anderson [30,31,32]. Their algorithm is based on a gravitational principle and calculates the velocity of a node based on summing the differences between the errors at other nodes and the mean error divided by the distance between the node and the other nodes. Since each node affects all other nodes, a global calculation is necessary to determine each node's speed in a computational grid.

Local mesh refinement is a different adaptive technique that consists of dividing or refining elements in regions where the solution is not adequately resolved. The advantage of this technique relative to mesh moving is that enough fine grids can be added to resolve the small scale structures of the solution and provide solutions to within user prescribed error tolerances. The local mesh refinement schemes of Berger [5,6,7], Flaherty and Moore [17], Gannon [20] and Bieterman and Babuska [8,9], have successfully satisfied prescribed error tolerances for different problems using finite element or finite difference schemes. The methods of Berger [5,6,7] and Gannon [20] have also been applied to two-dimensional problems.

The most promising algorithms appear to be those that combine both mesh moving and local mesh refinement. While neither technique nor their combination is likely to be optimal for all problems, a combination can accurately solve for the solution in regions where it varies rapidly and devote little effort in regions where it varies slowly. It is our intention to consider such schemes; however, the computational procedures discussed here do not as yet contain local refinement.

The mesh moving technique that we have developed is simple, efficient, and independent of the numerical method being employed to discretize the partial differential equations. At each time step it uses the current node locations and the nodal values of a mesh movement indicator. We use local error estimates or the solution gradients as mesh movement indicators. Nodes with "statistically significant error" (cf. Section 2) are grouped into rectangular error clusters. This clustering separates spatially distinct phenomena of the solution. As time evolves the clusters can move, change size, change orientation, collide, separate, reflect off boundaries, or pass through boundaries. At each time step new clusters can be created, and old ones can vanish. The clustering algorithms we use are briefly described in Section 2 and were developed by Berger [5,6] for a mesh refinement scheme for solving hyperbolic problems.

Mesh movement is determined by a node's relationship to the error clusters. Movement is done in two steps, each in a direction along a principal axis of a cluster rectangle. The amount of movement in each direction is determined by a movement function which insures that the center of error of the cluster moves according to a differential equation

suggested by Coyle et al. [11]. Additionally, the movement function smoothes mesh motion, reduces distortion and mesh tangling, and prevents nodes from moving outside the domain boundaries.

In Section 2 we discuss error clustering, movement of the center of mass of the error cluster, the node movement function, and the initial mesh generator used in the computational examples. In Section 3 we discuss the MacCormack finite difference scheme for hyperbolic equations and the error indicators used in the computational examples. The results of the computational examples are given in Section 4, and Section 5 contains a discussion of the results of the experiments and the status of our algorithm.

2. MESH MOVING SCHEME AND INITIAL MESH GENERATION

We discuss a mesh moving scheme and an initial mesh generator that can be used in conjunction with a numerical procedure to solve time dependent partial differential systems on a rectangular domain. Suppose that the domain is to be discretized into a moving mesh of quadrilateral cells having vertices (or nodes), $(x_i(t), y_i(t))$, $i = 1,2,\ldots,N$, that are numbered in a row sequential fashion. A sample mesh and a representative cell are shown in Figures 1 and 4, respectively. We further suppose that an approximate solution vector $u_i(t)$, $i = 1,2,\ldots,N$, of the partial differential system and a non-negative scalar error indicator, $e_i(t)$, $i = 1,2,\ldots,N$, are to be calculated at each node at time $t > 0$. The error indicator can be related to the local discretization error at a node; however, quantities proportional to the solution gradient, curvature, etc., can also be used. The error indicator serves to attract

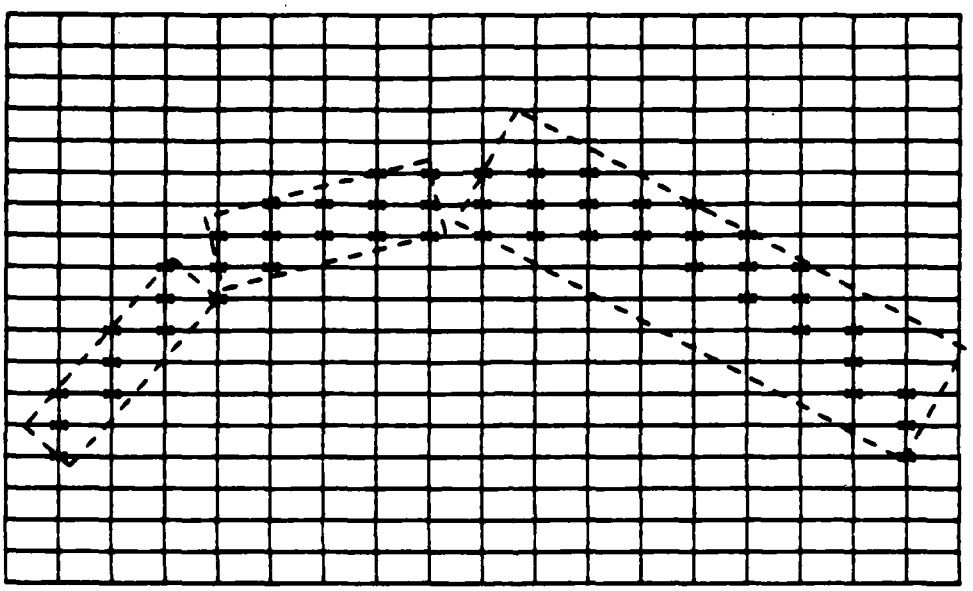


Figure 1. Three clusters of significant error aligned with a curved significant error region.

nodes, and thus, it should be large where the mesh should be fine and small where the mesh should be coarse. The mesh moving scheme is discussed first in Section 2.1 and the discussion of the initial mesh generator follows in Section 2.2.

2.1. MESH MOVING SCHEME

Suppose the mesh, solution, and error indicator described above have been calculated up to a time $t > 0$. We scan the mesh at time t and flag "significantly high error nodes" as nodes having error indicators greater than twice the mean nodal error indicator and also greater than a user supplied error indication tolerance. If there are no significant error nodes, computation is performed on a stationary mesh. The nearest neighbor clustering algorithm of Berger [5,6] is used next to cluster the flagged error nodes. In this iterative algorithm, a cluster is first defined to consist of one arbitrary significantly high error node. Other significantly high error nodes are added to the cluster if they are within a specified minimum intercluster distance from the nearest node in the cluster. New clusters are established for nodes that do not belong to any existing cluster. Clusters are united when a node is determined to belong to more than one of them. Upon completion of the algorithm, (i) nodes in different clusters will be separated by at least the minimum intercluster distance, and (ii) no node in a cluster with more than one node will be further than the minimum intercluster distance from its nearest neighbor in the cluster.

Berger [5,6] shows that near minimum area rectangles that contain a cluster can be easily generated. The principal axes of such a rectangle

are the major and minor axes of an enclosed ellipse with the same first and second moments as the clustered nodes. Thus, if x_m and y_m are the mean coordinates of the clustered nodes, then the axes of the rectangle are in the directions of the eigenvectors of the symmetric (2x2) matrix

$$\begin{bmatrix} \Sigma\,(x_i^2 - x_m^2) & \Sigma\,(x_i\,y_i - x_m y_m) \\ \Sigma\,(x_i y_i - x_m\,y_m) & \Sigma\,(y_i^2 - y_m^2) \end{bmatrix} \qquad (2.1)$$

The summations range over all nodes in the cluster.

For problems with significant error nodes located on a long curved arc, the entire region may belong to one unacceptably large cluster. In order to prevent this inefficieny and provide better alignment with curved fronts, the rectangular clusters are checked for efficiency by determining the percentage of significant error nodes in the cluster. If a 50 percent efficiency is not achieved, the rectangle is iteratively bisected in the direction of the major axis. This is repeated until all clusters have a 50 percent efficiency or more. This nearest neighbor clustering separates spatially distinct phenomena as shown by the dashed line error clusters on the two dimensional mesh of Figure 14 and provides some linear alignment with long curved error regions as shown by the clusters in Figure 1.

We determine node movement from the velocity of propagation, the orientation, and the size of the error clusters. Thus, our approach differs from that of Hyman [25] and Harten and Hyman [22] who move

nodes so as to minimize the time variation of the solution components. For hyperbolic systems their approach allows the mesh to move at a weighted average of the characteristic speeds along a shock. Front tracking schemes also move the mesh so that isolated discontinuities are stationary in reference to the mesh. Since clustering significantly high error nodes and tracking the propagation of these clusters is possible for all time dependent problems, our approach is more general and approximates these other algorithms for hyperbolic problems. We do not, however, follow or track surfaces of discontinuities exactly. We assume that nodes in the same cluster have related solution characteristics, so that we can determine individual node movement from the propagation of the center of mass of the error cluster.

In the Harten and Hyman [22] algorithm, when there is multiple wave interaction in a vector system, mesh speed is a weighted average of the characteristic velocities. The same principle applies to our algorithm when multiple error clusters have merged due to, e.g., wave interaction. The mesh moves with a velocity given by a weighted average of the velocities of the intersecting error clusters. Comparisons between the center of mass propagation of an error cluster and the characteristic path of the center of the cluster are given in Example 4.2.

We first attempted to move nodes using a procedure that was based on extrapolation of the positions of the previous center of error masses; however, this produced some oscillatory effects. Indeed, Coyle et al. [11] showed that node movement based on extrapolation is unstable in certain situations. Following one of their suggestions, we stablize the movement by solving the differential equation

$$\ddot{r} + \lambda \dot{r} = 0 , \qquad (2.2)$$

where r(t) is the position vector of the center of mass of an error cluster and $\dot{(\ )} := d(\ )/dt$. Equation (2.2) is conditionally stable (cf. Coyle et al. [11]), and when solved numerically with reasonable choices of $\lambda > 0$ oscillations in the mesh motion were no longer present.

We solve (2.2) from t_{n-1} to t_n and then for each cluster determine $r(t_{n+1})$ and the vector $r(t_{n+1}) - r(t_n)$ which is projected onto the two principal axial directions of the rectangular cluster at t_n. These projected distances are the amount by which the center of mass of the error cluster moves in each principal direction. Let Δr_1 and Δr_2 denote these projections, respectively, and let CM denote the center of mass of the error cluster. We create a one-dimensional mesh movement function to move the nodes of the mesh along the two axial directions of the error clusters. A profile of the movement function that we use is shown in Figure 2; however, the algorithm is designed to be used with any one dimensional movement function. The slope of the movement function depends on the length of the side of the cluster which is denoted as w in Figures 2 and 3.

As shown in Figure 2, nodes inside the range of the cluster (shaded in Figure 3) are moved a distance $d_{i,inside}$, i = 1,2, in each principal direction given by

$$d_{i,inside} = \begin{cases} \Delta r_i(3/2 - x/w) & \text{if } w/2 \le x \le 3w/2 \\ \Delta r_i & \text{if } -w/2 < x < w/2 \\ \Delta r_i(3/2 + x/w) & \text{if } -3w/2 \le x \le -w/2, \end{cases}$$

$$i = 1,2. \qquad (2.3)$$

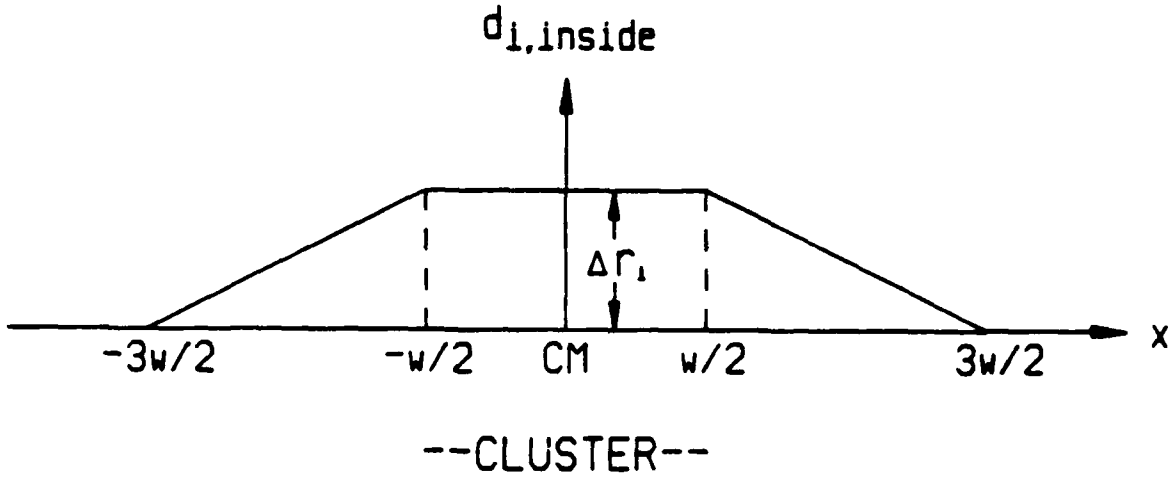


Figure 2. Profile of the node movement function (cf. Eq. (2.3))

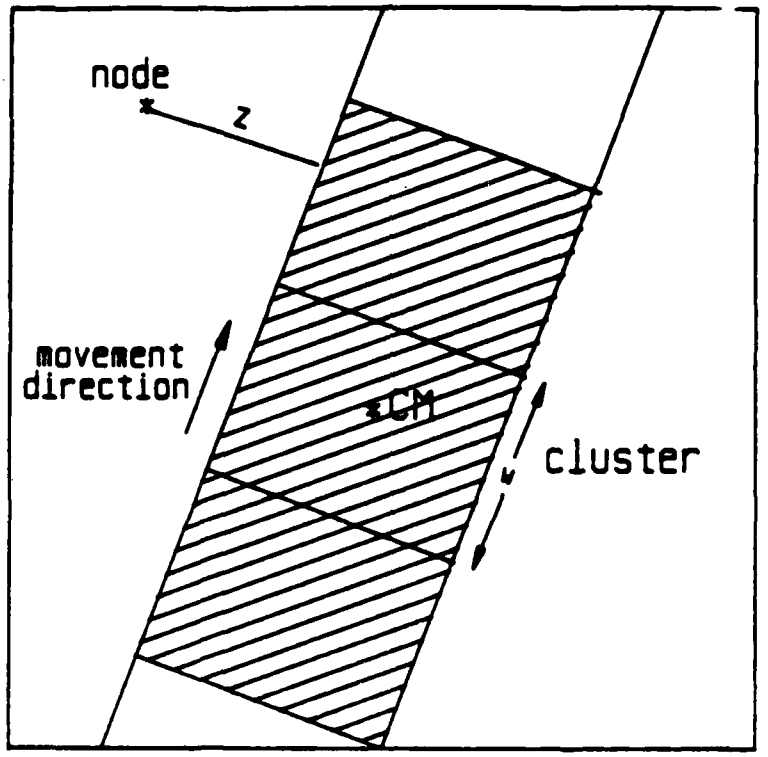


Figure 3. A node shown outside the range of an error cluster. The distance z is used in Equation (2.4) to determine the amount of movement for this node.

Here x is the projected distance in a principal direction of a node relative to the center of mass of the error cluster. In order to provide smooth node movement throughout the domain, nodes outside the range of the cluster move in a reduced amount as determined by

$$d_{i,outside} = d_{i,inside}[\ 1-(2z/D)]\ , \qquad (2.4)$$

where z is the shortest distance to the range of the cluster (cf. Figure 3) and D is the diagonal distance of the entire domain. Node movement distances $d_{i,inside}$ and $d_{i,outside}$ are reduced near boundaries in order to prevent nodes from leaving the domain. In particular, for nodes moving towards the edge of the domain, we recalculate $d_{i,j}$ as $d_{i,j}$ [min(1,b/c)], i = 1,2, j = inside, outside, where b is the distance of the node to the boundary and c is twice the length of a cell diagonal on a uniform mesh having the same number of cells as the moving mesh. Nodes on domain boundaries, except corner nodes which are not moved, are restrained to move along the boundary.

2.2. INITIAL MESH GENERATION

The generation of a proper initial mesh is critical to the success of the mesh moving scheme. Without refinement the mesh moving algorithm cannot provide suitable error control unless the initial mesh spacing properly resolves initial data. An initial error measure appropriate for the finite difference scheme on quadrilateral cells of Section 3 is the error in interpolating the prescribed initial condition $u_0(x,y)$ on each cell by a bilinear polynomial. The error on each cell is determined as the difference between the value of the initial function and its bilinear interpolant at the center of each cell. Therefore, the initial mesh must

be generated so that the condition

$$|1/4\{u_0(x_i,y_i)+u_0(x_j,y_j)+u_0(x_k,y_k)+u_0(x_l,y_l)\}-u_0(\bar{x},\bar{y})| < TOL \qquad (2.5)$$

holds on each cell when using the vertex and center point labelling as shown for a general cell in Figure 4. TOL is a user supplied error tolerance. We satisfy condition (2.5) using an iterative scheme that begins by generating a uniform mesh and computing an initial error estimate from the left side of (2.5), we cluster nodes of high error and move them toward centers of the clusters, and recompute the initial error. Then we iteratively add rows and columns to the mesh wherever the error tolerance is exceeded, smooth the mesh by the scheme of Brackbill and Saltzman [10], and recompute the error until the user tolerance is satisfied.

Initial meshes generated with this algorithm are shown in Figures 5, 7, 14, and 18 for the initial conditions of the computational examples of Section 4. Any initial mesh generator that satisfies condition (2.5) could be used in this scheme, and several such algorithms can be found in Thompson [34].

3. MacCORMACK FINITE DIFFERENCE SOLVER AND ERROR INDICATION

In order to test our mesh moving scheme, we used the explicit finite difference MacCormack scheme on nonuniform quadrilateral grids for hyperbolic vector systems of conservation laws having the form

$$u_t + f_x(x,y,u,t) + g_y(x,y,u,t) = 0, \qquad (3.1)$$

$$u(x,y,0) = u_0(x,y), \qquad (3.2)$$

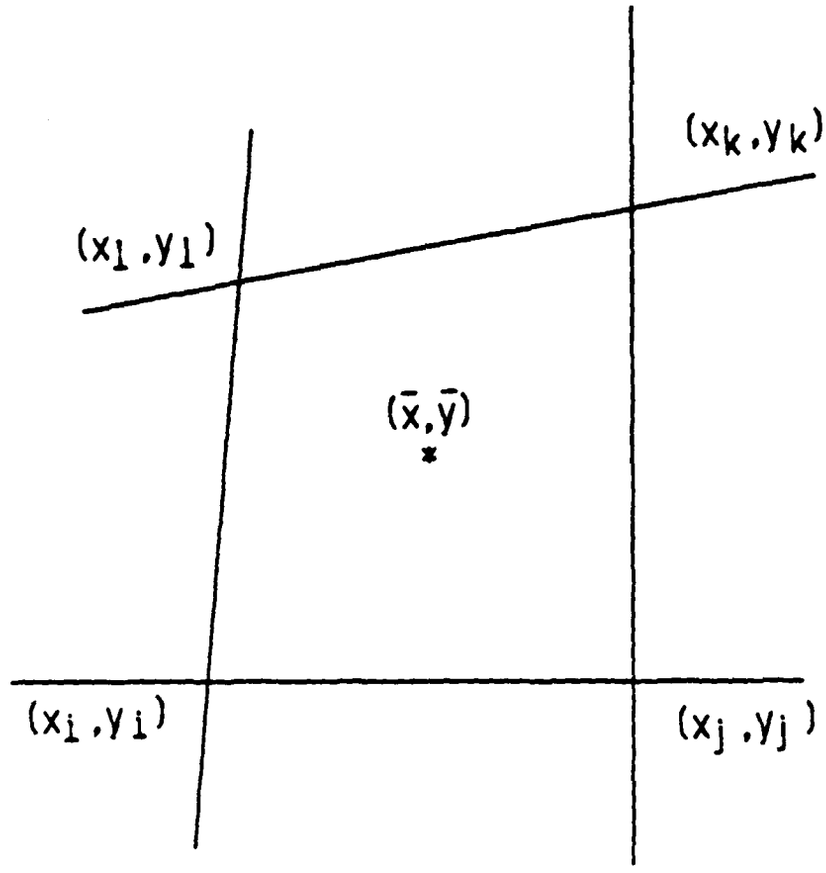


Figure 4. Node labelling of an arbitrary quadrilateral cell.

with appropriate well posed boundary conditions.

In order to discretize (3.1) we introduce a transformation

$$\xi = \xi(x,y,t), \qquad \eta = \eta(x,y,t), \qquad \tau = t, \tag{3.3}$$

from the physical (x,y,t) domain to a computational (ξ,η,τ) domain where a uniform rectangular grid will be used. Under this transformation (3.1) becomes

$$u_\tau + u_\xi \xi_t + u_\eta \eta_t + f_\xi \xi_x + f_\eta \eta_x + g_\xi \xi_y + g_\eta \eta_y = 0 . \tag{3.4}$$

The transformation metrics $(\xi_x, \xi_y, \xi_t, \eta_x, \eta_y, \eta_t)$ are related to the metrics $(x_\xi, x_\eta, x_\tau, y_\xi, y_\eta, y_\tau)$ of the inverse mapping of the computational domain to the physical domain by the identities

$$\begin{aligned} \xi_x &= y_\eta / J , \quad \xi_y = -x_\eta / J , \quad \xi_t = (y_\tau x_\eta - x_\tau y_\eta) / J, \\ \eta_x &= -y_\xi / J , \quad \eta_y = x_\xi / J , \quad \eta_t = (y_\xi x_\tau - x_\xi y_\tau) / J , \\ J &= y_\eta x_\xi - x_\eta y_\xi . \end{aligned} \tag{3.5}$$

Using (3.5) in (3.4) gives

$$\begin{aligned} &u_\tau + u_\xi (y_\tau x_\eta - x_\tau y_\eta) / J + u_\eta (y_\xi x_\tau - x_\xi y_\tau) / J \\ &+ f_\xi y_\eta / J + f_\eta (-y_\xi / J) + g_\xi (-x_\eta / J) + g_\eta x_\xi / J = 0 . \end{aligned} \tag{3.6}$$

We discretized (3.6) by the MacCormack scheme using first order forward difference approximations in the predictor and first order backward differences in the corrector step. It was shown by Hindman [23,24] that this differencing of Equations (3.4) or (3.6) produces consistent and

conservative approximations. We automatically adjust the time step so as to satisfy the Courant, Friedrichs, Lewy Theorem.

Accurate error estimation is important to insure that user tolerances are achieved and to refine proper regions when doing local mesh refinement. However, mesh moving techniques are not as sensitive as refinement techniques to error estimation. As long as the error indicator shows the important error features and propagation characteristic, proper error magnitudes are not necessary. Therefore, in the computational examples of Section 4, we were able to use either the solution gradients or the difference between the predicted solution and the corrected solution as the error or movement indicator. This error indication is actually an error estimation for the first order predicted solution and not the second order corrected solution; however, it does have the proper propagation characteristics. Other more reliable error estimates will be needed when local mesh refinement is introduced. Error estimators that we are investigating are based on combining extrapolation and the difference between the predicted and corrected solutions. Results of this method will be reported in Arney [38]. Other possibilities are to use hierarchical approximations as done by, e.g., Adjerid and Flaherty [1] and Zienkiewicz et al. [37].

4. COMPUTATIONAL EXAMPLES

The following hyperbolic equations were solved using the initial mesh generator of Section 2 and the integrator of Section 3 as tests of our mesh moving technique. Plots of the grids show significant error nodes marked with an asterisk and the rectanglar error clusters outlined

with dashed lines.

Example 4.1. Consider the linear scalar hyperbolic differential equation

$$u_t + u_x + 0.25u_y = 0, \qquad t > 0\ ,\ 0.2 \le x \le 1.2\ ,\ 0 \le y \le 1, \quad (4.1)$$

with initial conditions

$$u(x,y,0) = \begin{cases} 0\ , & \text{if } y < -4x + 1.2 \\ 0.8\ , & \text{if } y > -4x + 1.6 \\ -8x - 2y + 3.2\ , & \text{otherwise,} \end{cases} \quad (4.2)$$

and with Dirichlet boundary conditions

$$u(x,y,t) = \begin{cases} 0\ , & \text{if } y - 0.25t < -4(x - t) + 1.2 \\ 0.8\ , & \text{if } y - 0.25t > -4(x - t) + 1.6 \\ -8(x - t) - 2(y - 0.25t) + 3.2\ , & \text{otherwise.} \end{cases} \quad (4.3)$$

The solution of this problem is an oblique front that moves at an angle of 14 degrees across the domain. We selected this problem to show the concentration of nodes of the initial mesh within the front, the partial alignment of the initial mesh with the front, the propagation of the refined region of the mesh with the moving front, and the reduction of dispersive errors with a moving mesh. The movement indicator used in this example is the magnitude of the gradient of the solution.

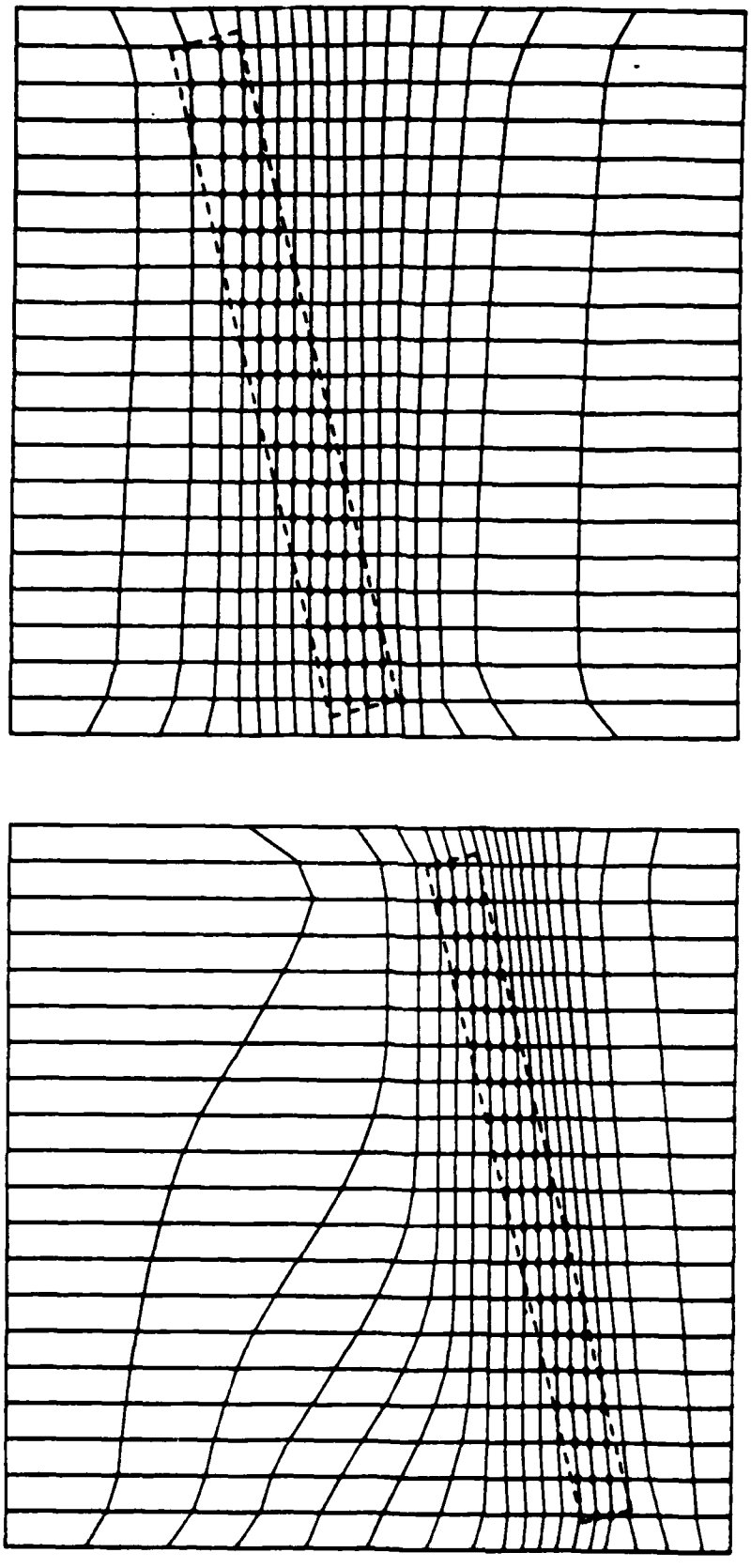


Figure 5. Meshes of Example 4.1 at t = 0.0 (top) and at t = 0.4 (bottom)

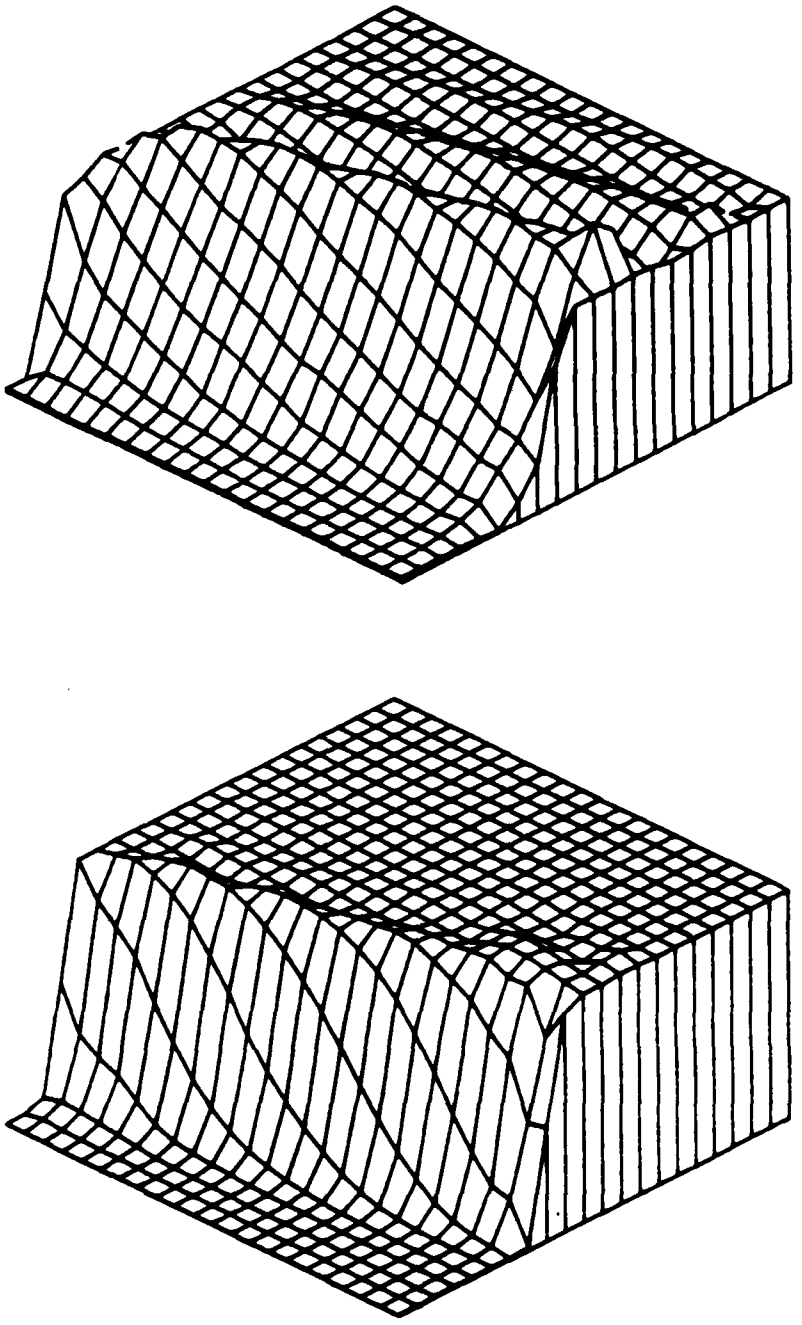


Figure 6. Surface plots of the solution of Example 4.1 at t = 0.4 using a stationary uniform mesh (top) and a moving mesh (bottom). The moving mesh reduces the numerical oscillations behind the propagating wave.

The computational meshes at t = 0 and t = 0.4 are shown in Figure 5. A value of .01 was used for TOL in Eq. (2.5). At each timestep the nodes are moving with the wave front at near the characteristic speed. This reduces dispersive errors as shown in the surface plots of Figure 6 which compare the solution calculated on the moving mesh to one calculated on a stationary uniform mesh having the same number of nodes. Observe that node movement occurs without severe distortion of cells or nodes accumulating near, or passing through, the outflow boundaries.

Example 4.2. Consider the initial-boundary value problem

$$u_t - yu_x + xu_y = 0, \quad t > 0\ ,\ -1.2 \le x \le 1.2,\ -1.2 \le y \le 1.2, \tag{4.4}$$

$$u(x,y,0) = \begin{cases} 0, & \text{if } (x-1/2)^2 + 1.5\,y^2 \ge 1/16 \\ 1 - 16((x-1/2)^2 + 1.5y^2), & \text{otherwise}\ , \end{cases} \tag{4.5}$$

and

$$u(1.2,y,t) = u(-1.2,y,t) = u(x,-1.2,t) = u(x,1.2,t) = 0\ . \tag{4.6}$$

The exact solution of this problem is

$$u(x,y,t) = \begin{cases} 0, & \text{if } C < 0 \\ C, & \text{if } C \ge 0, \end{cases} \tag{4.7}$$

where

$$C = 1 - 16((x\cos t + y\sin t - 1/2)^2 + 1.5(y\cos t - x\sin t)^2). \tag{4.8}$$

Equations (4.7) and (4.8) represent a moving elliptical cone rotating counterclockwise around the origin with period 2π. This problem was proposed as a test problem by Gottlieb and Orszag [17] and we selected it because the rotational quality of the error region is a good test of a mesh moving scheme.

The initial mesh generated for this problem is shown in Figure 7. This mesh has an initial interpolation error less than 0.08. Figures 8 and 9 show the mesh at $t = 1.6$ and $t = 3.2$, respectively. The nodes follow the moving cone and keep it within the refined region. Figures 10 and 11 compare the contour and surface plots, respectively, of the solution at $t = 3.2$ on the moving mesh with one on a 32 x 32 uniform stationary mesh. The dispersive error distorts the cone and leaves a wake behind it. These errors are significantly reduced by the mesh moving solution.

Figure 12 compares the path of the center of mass of the single error cluster using (2.2) and the real characteristic path of the peak of the cone. As expected for this scalar hyperbolic problem, the movement of the center of error mass determined by (2.2) closely approximates the characteristic path of the peak of the cone with a maximum difference of 4 percent in length and direction.

Example 4.3. This problem is similiar to Example 4.2 except there are now two symmetric cones rotating counterclockwise about the origin. The problem is given by Equations (4.4), (4.6), and new initial conditions provided by

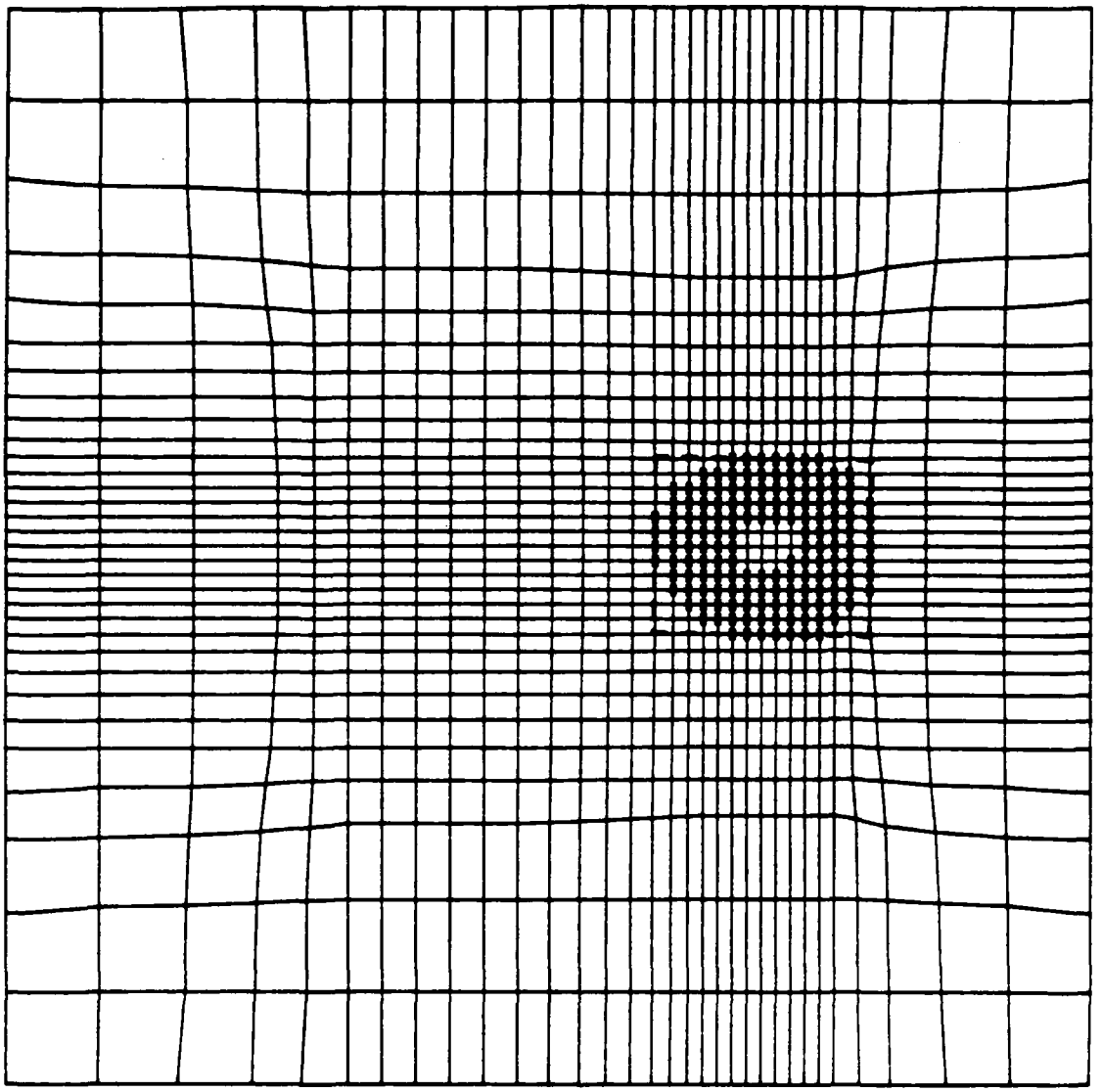


Figure 7. Initial mesh for the rotating cone of Example 4.2.

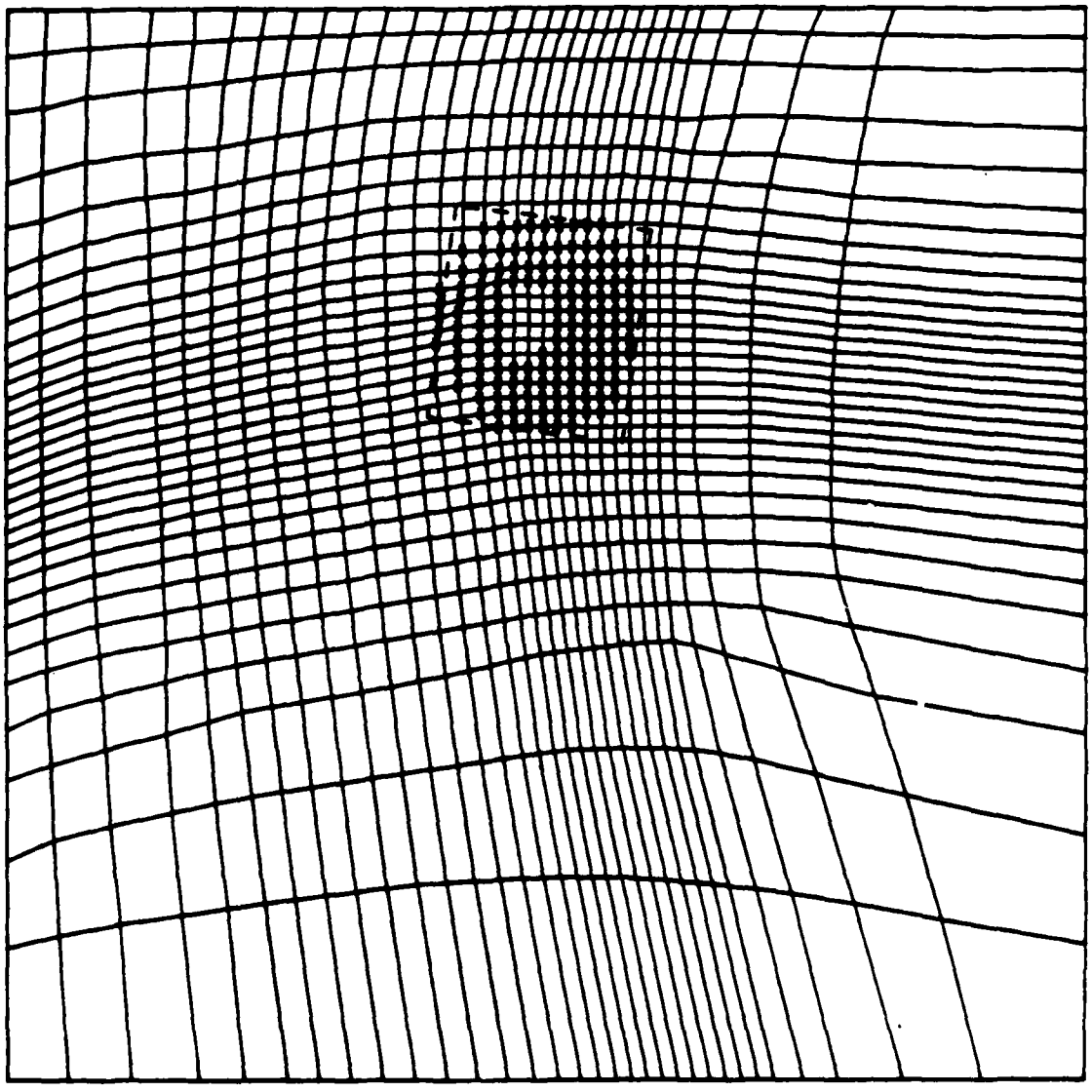


Figure 8. Mesh of Example 4.2 at t = 1.6. Nodes are moving with the rotating cone.

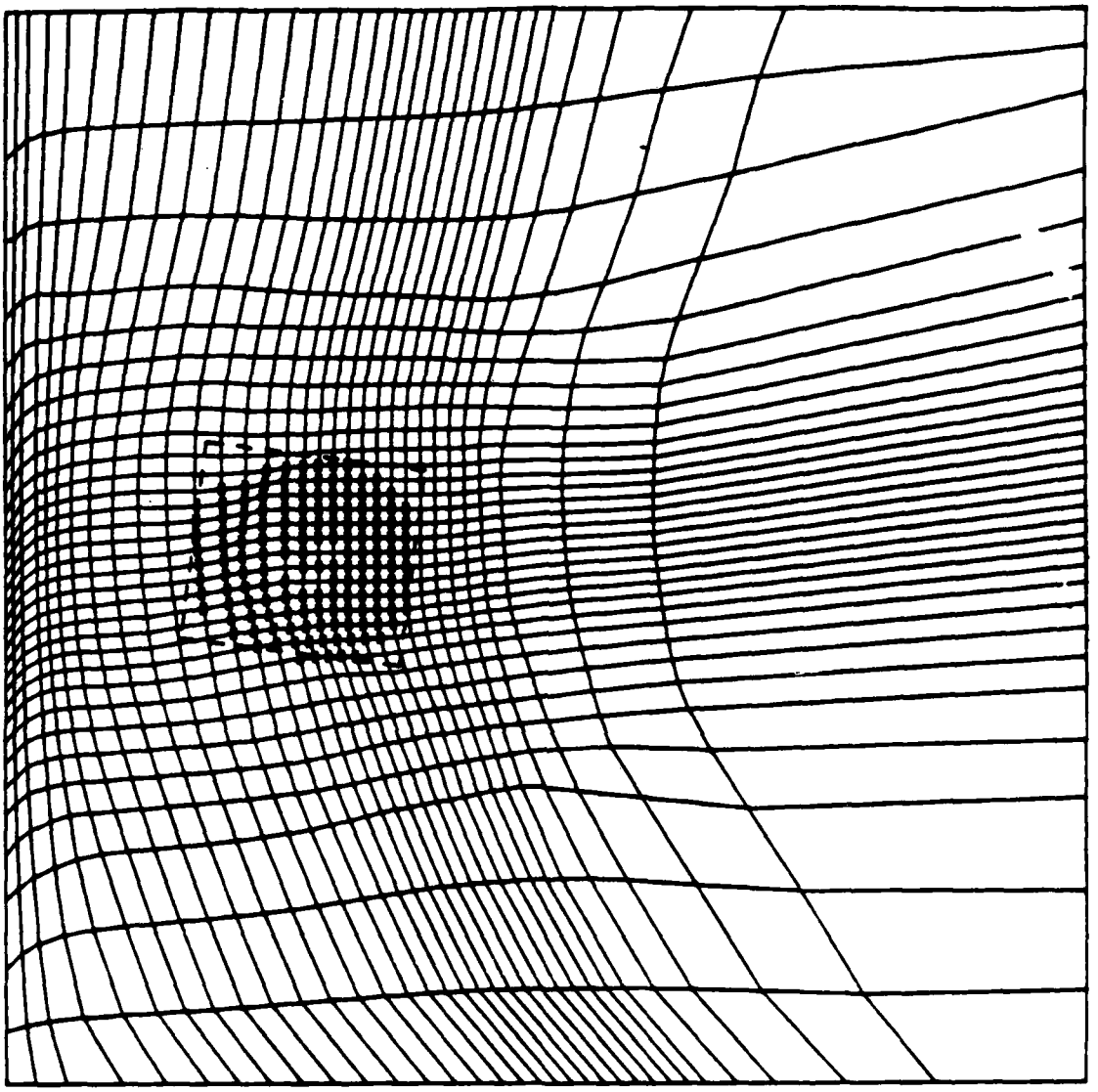


Figure 9. Mesh of Example 4.2 at t = 3.2. Nodes continue to move with the rotating cone with some node crowding near the boundary.

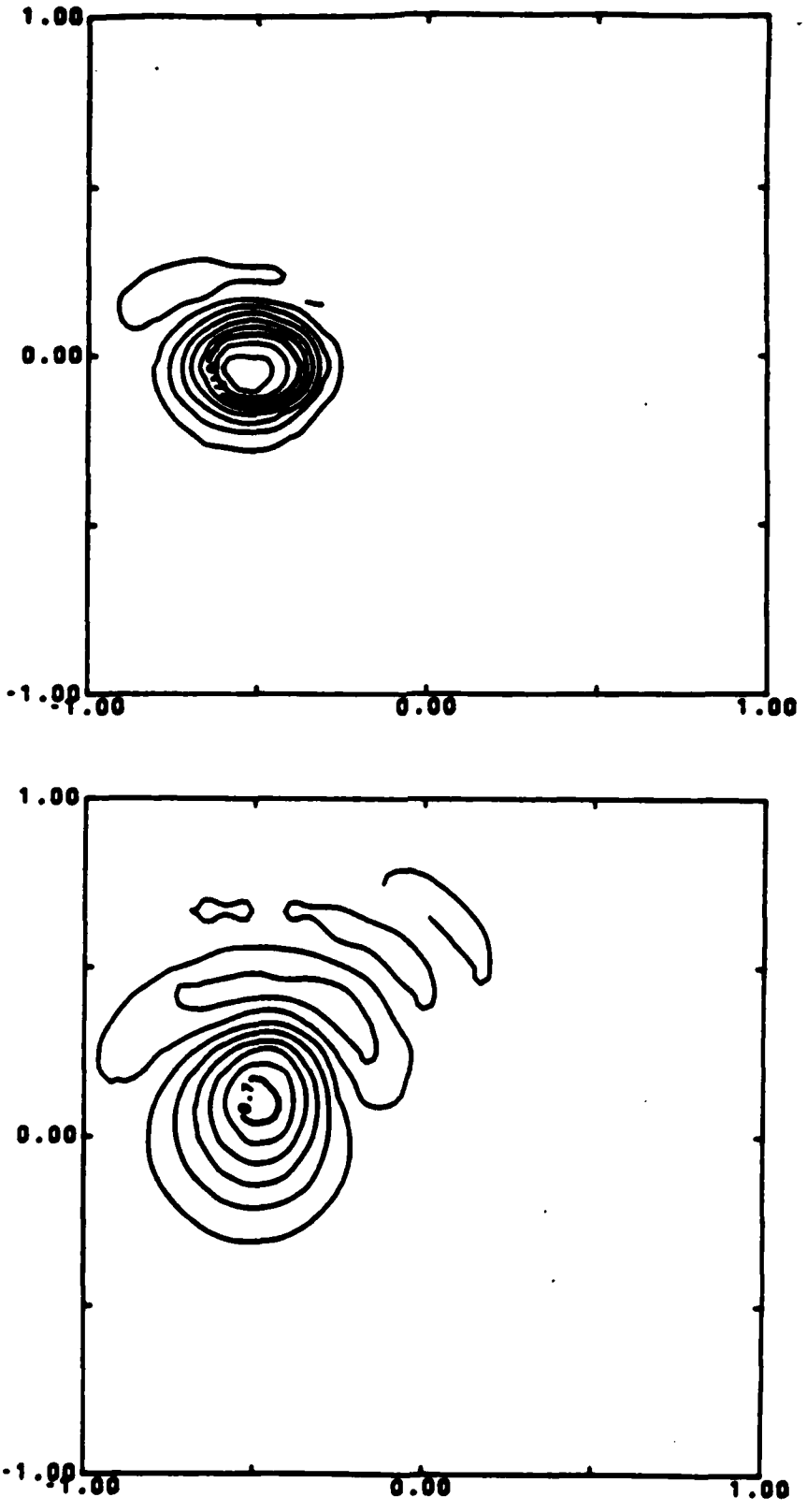


Figure 10. Contour plots of solutions of Example 4.2 on a moving mesh (top) and on a stationary uniform mesh (bottom) at t = 3.2.

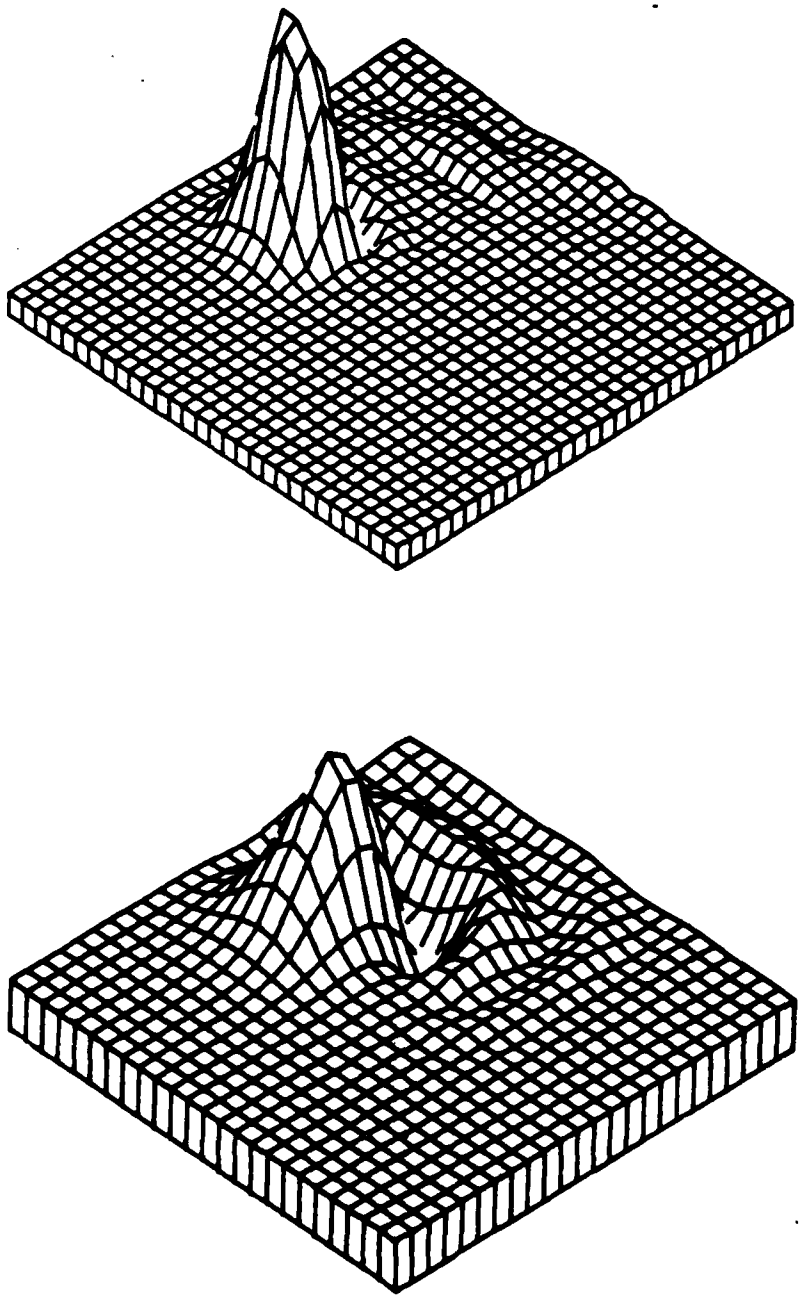


Figure 11. Surface plots of solutions of Example 4.2 on a moving mesh (top) and on a stationary uniform mesh (bottom) at t = 3.2.

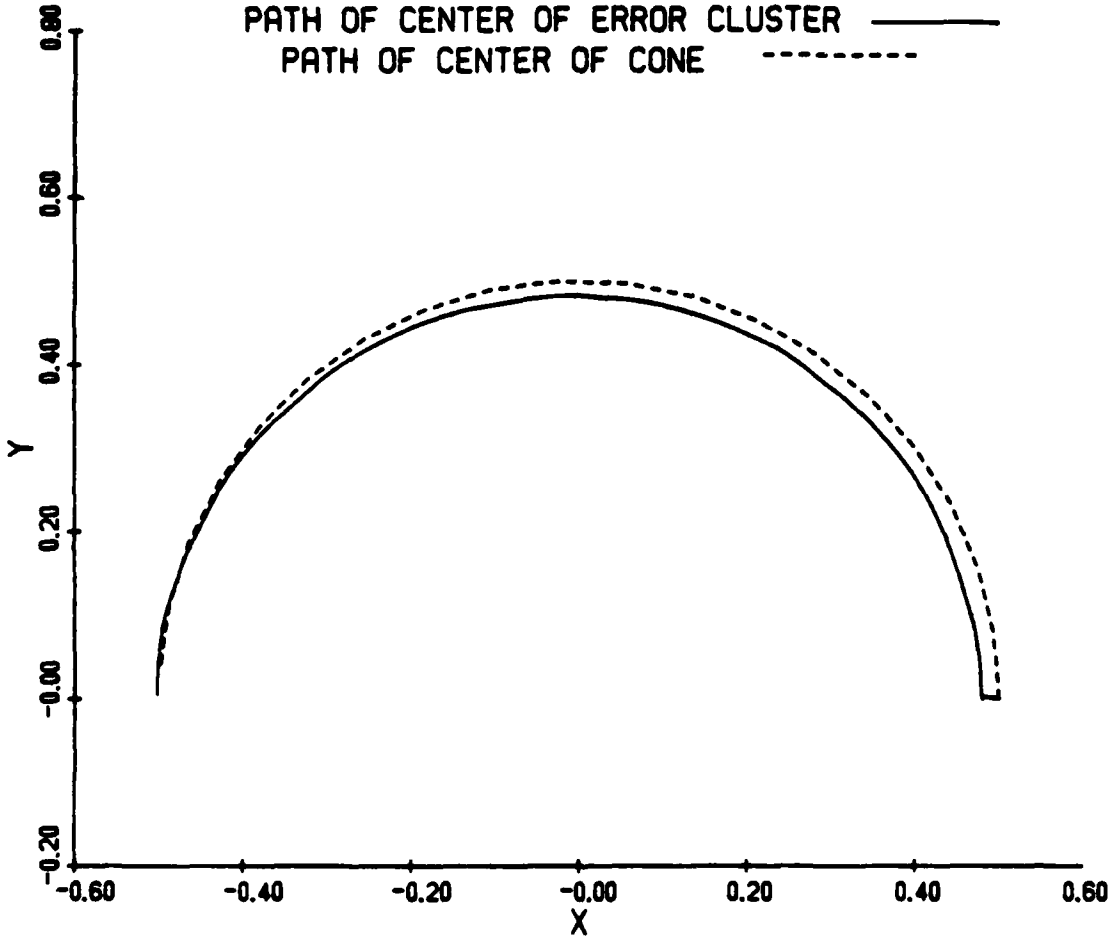


Figure 12. Comparison of the characteristic path of the peak of the cone and the path of the center of error mass as determined by Equation (2.2) for Example 4.2.

$$u(x,y,0) = \begin{cases} 1 - 16((x-1/2)^2 + 1.5y^2), & \text{if } (x-1/2)^2 + 1.5y^2 \le 1/16 \\ 1 - 16((x+1/2)^2 + 1.5y^2), & \text{if } (x+1/2)^2 + 1.5y^2 \le 1/16 \\ 0, & \text{otherwise.} \end{cases} \quad (4.9)$$

Figure 13 shows the mesh at t = 1.05, which has poor aspect ratios and severe mesh distortion caused by the rotation of the error regions. The mesh tangles as the cones rotate further. When such mesh tangling occurs a static rezone is necessary to create a new mesh. The rezoning can use an algorithm similar to the one that generated the initial mesh. The data at the new mesh nodes must be obtained by interpolation from the calculated solution at the old nodes by a conservative rezoning as presented in Dukowicz [14]. We are exploring this possibility.

Example 4.4. Consider the uncoupled system

$$\begin{aligned} &u_t + u_x = 0, \\ &\qquad t > 0,\ -1 \le x \le 1,\ -1 \le y \le 1, \qquad (4.10) \\ &v_t - v_x = 0, \end{aligned}$$

$$v(x,y,0) = \begin{cases} 1 - 16((x+1/2)^2 + 1.5y^2), & \text{if } (x+1/2)^2 + 1.5y^2 \le 1/16 \\ 0, & \text{otherwise,} \end{cases} \quad (4.11a)$$

$$u(x,y,0) = \begin{cases} 1 - 16((x-1/2)^2 + 1.5y^2), & \text{if } (x-1/2)^2 + 1.5y^2 \le 1/16 \\ 0, & \text{otherwise,} \end{cases} \quad (4.11b)$$

and

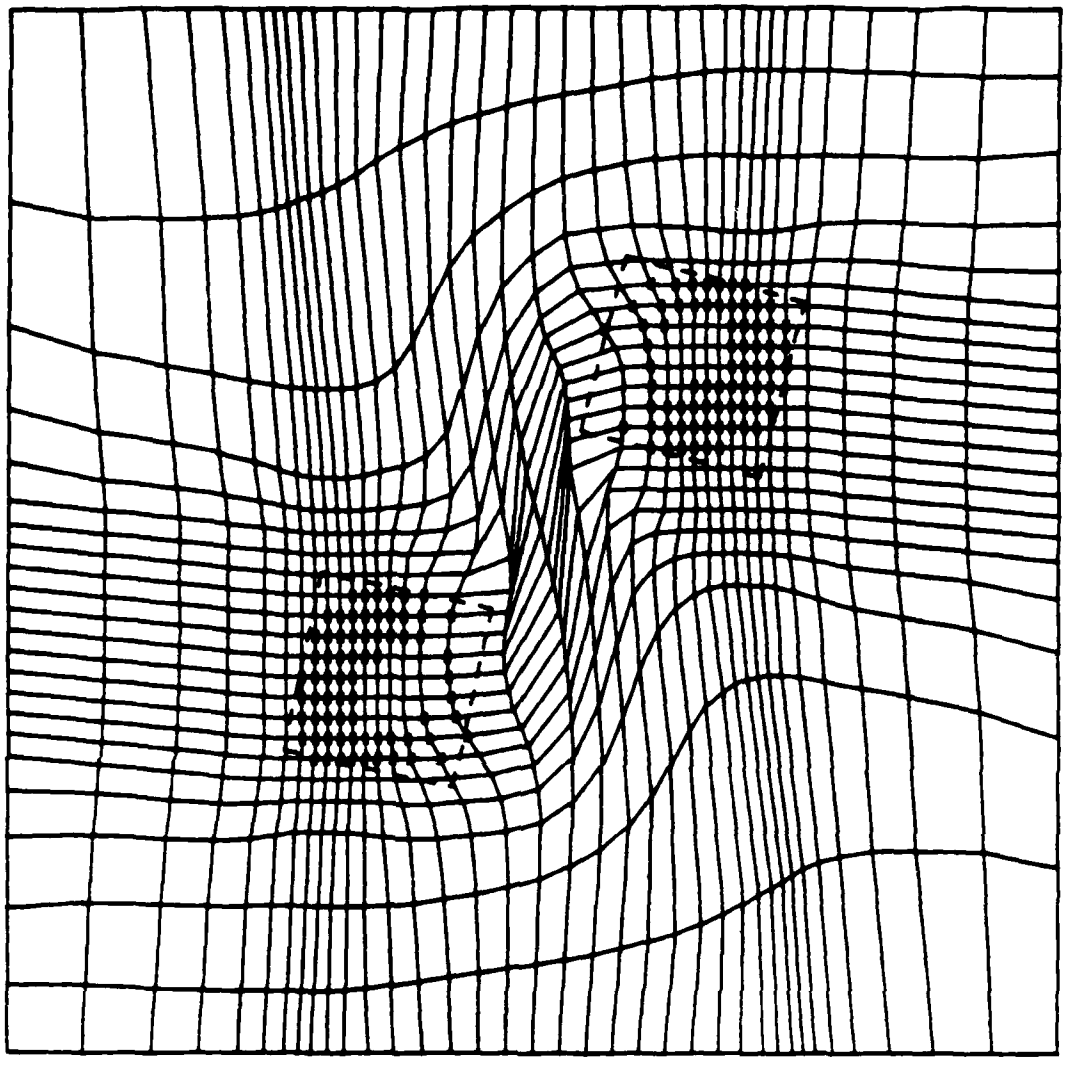


Figure 13. Distorted mesh of Example 4.3 at t = 1.05 showing the need for static rezoning.

$$u(x,y,t) = v(x,y,t) = 0 \text{ on all boundaries of the domain.} \qquad (4.12)$$

The solution of this problem is two moving cones that pass through one another. This causes the error clusters to collide and merge, and then later separate. Figure 14 shows the initial mesh for this problem, and Figure 15 shows the mesh at t = 0.35, the time at which the clusters collided and merged. From t = 0.35 to t = 0.9, the single cluster stays centered at the origin so the mesh does not move during this time. At t = 0.9 the cones have passed completely through one another, and Figure 16 shows the separation of the error clusters and the movement of the mesh toward the boundaries at that time. Figure 17 shows the mesh at t = 1.3. The cones and error clusters have reached the domain boundary and no further movement of the mesh will take place as the cones exit the domain.

Example 4.5. Consider the Euler equations for a perfect inviscid fluid

$$Q_t + E_x + F_y = 0 \qquad (4.13a)$$

where

$$Q = \begin{bmatrix} \rho \\ \rho u \\ \\ \rho v \\ e \end{bmatrix} \quad E = \begin{bmatrix} \rho u \\ \rho u^2 + p \\ \\ \rho uv \\ (e + p)u \end{bmatrix} \quad F = \begin{bmatrix} \rho v \\ \rho uv \\ \\ \rho v^2 + p \\ (e + p)v \end{bmatrix}. \qquad (4.13b)$$

In the above equations, u and v are the velocity components in the x and y directions, ρ is the density, e is the total energy per unit volume, and p is the pressure which is given by

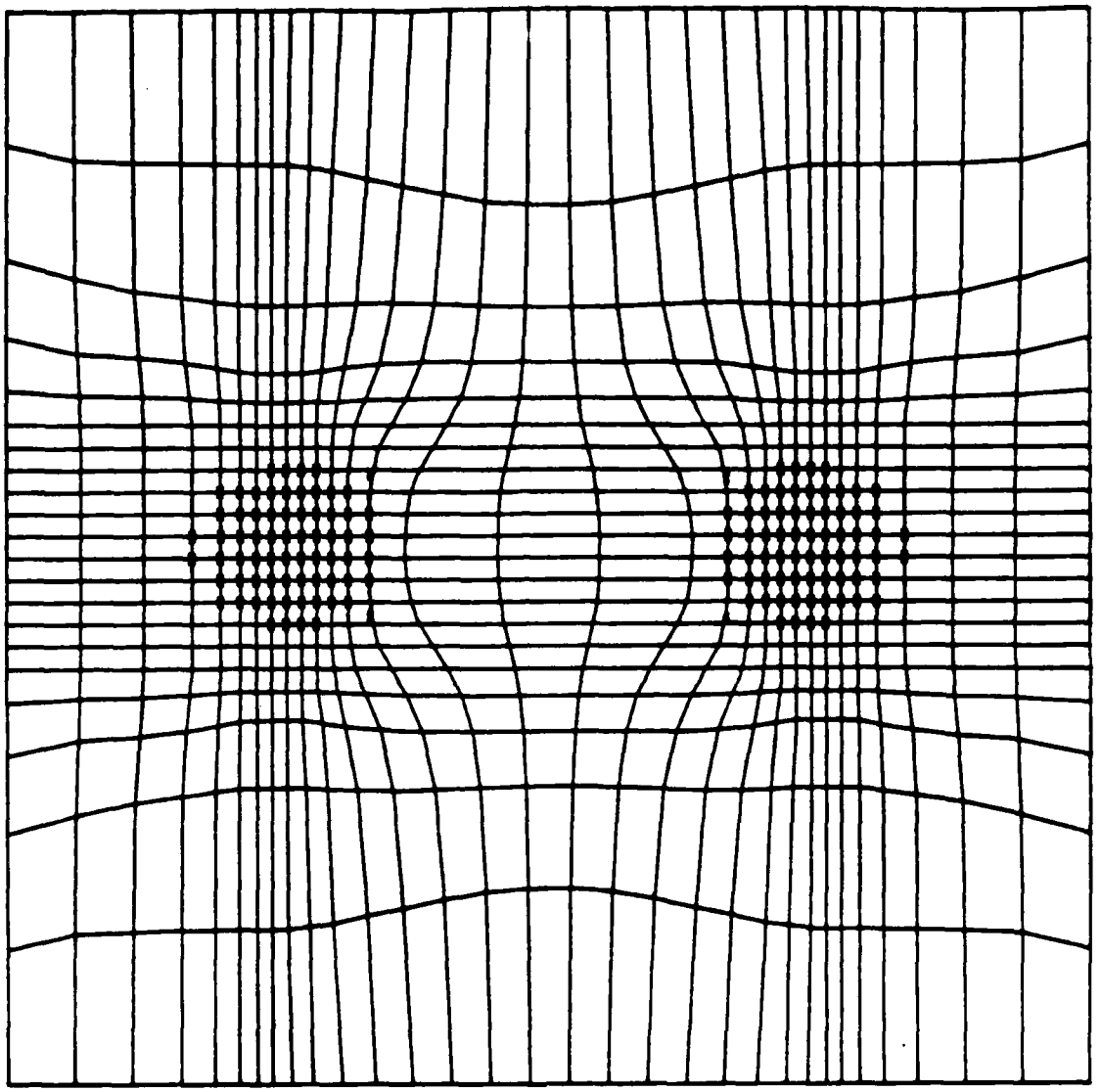


Figure 14. Two spatially distinct error clusters formed by the nearest neighbor clustering algorithm for the initial mesh of Example 4.4

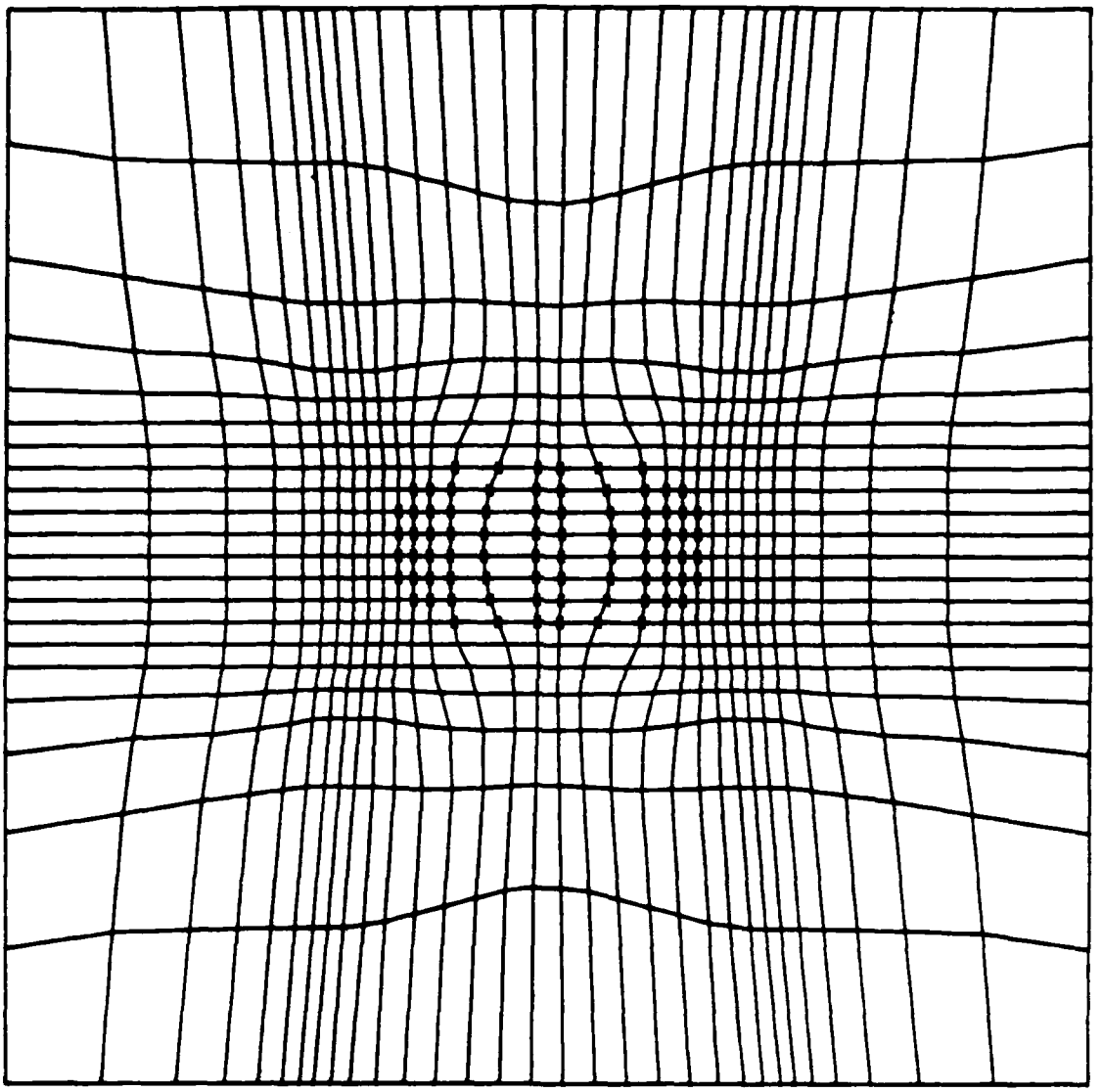


Figure 15. Mesh of Example 4.4 at t = 0.35. The two initial clusters have merged into single cluster centered at the origin.

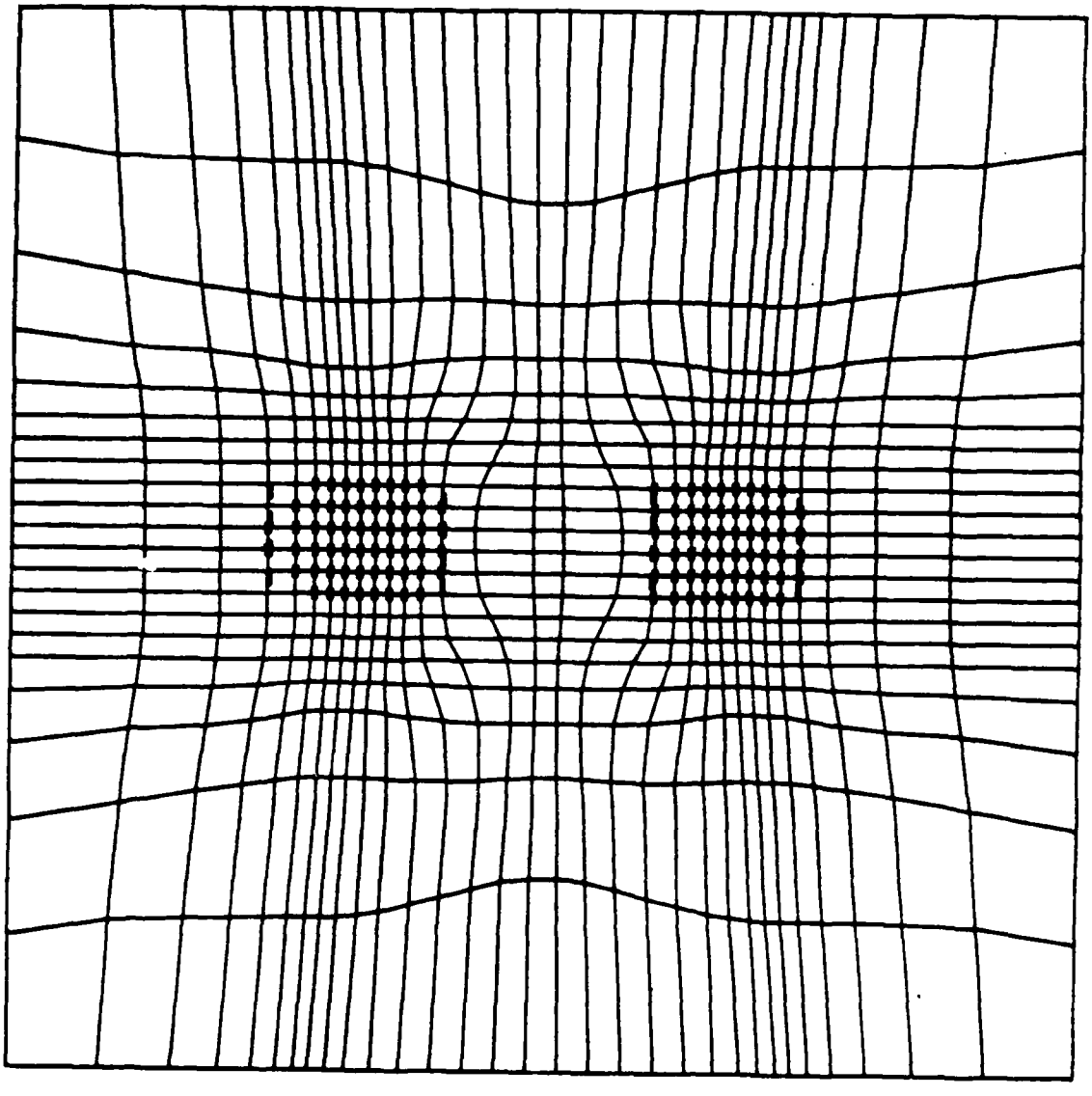


Figure 16. Mesh of Example 4.4 at t = 0.9. The single cluster separated into two clusters. The two clusters are moving toward the domain boundaries.

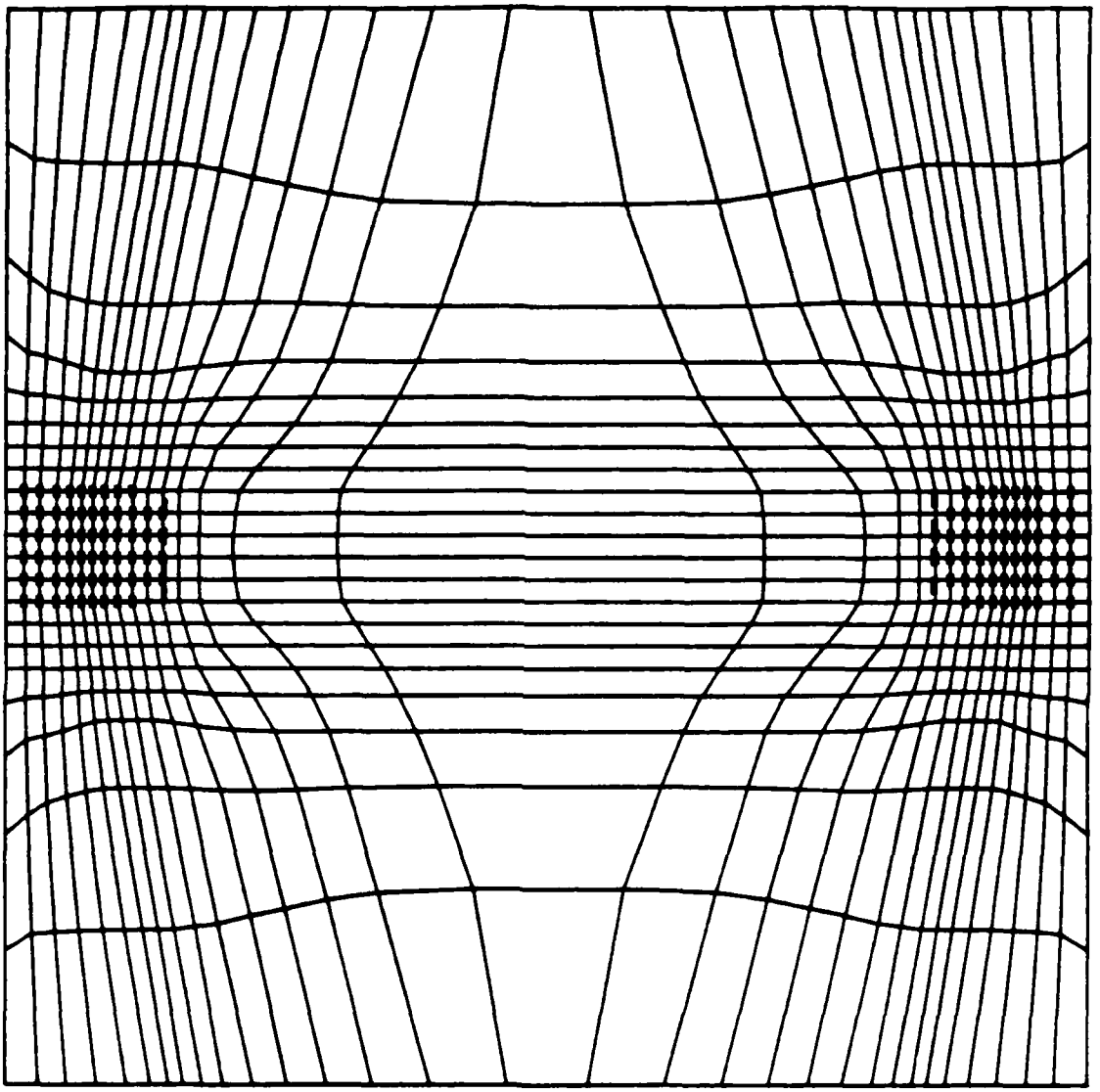


Figure 17. Mesh of Example 4.4 at t = 1.4. The two clusters have reached the domain boundaries.

$$p = (\gamma - 1)[e - \rho(u^2 + v^2)/2]. \qquad (4.13c)$$

We solve a problem where a Mach 10 shock moves down a channel containing a wedge. The computational domain is $-0.3 \le x \le 1.9$, $0 \le y \le 1$, which is oriented along the wedge so that the wedge lies on the bottom boundary in the region $y = 0$, $1/6 \le x \le 1.9$. The initial conditions

$$\rho = 8.0,\ p = 116.5,\ e = 563.5,\ u = 4.125\sqrt{3},\ v = -4.125,$$
$$\text{if } y < \sqrt{3}\,x - \sqrt{3}/6\ , \qquad (4.14a)$$

and

$$\rho = 1.4,\ p = 1.0,\ e = 2.5,\ u = 0.0,\ v = 0.0,$$
$$\text{if } y \ge \sqrt{3}\,x - \sqrt{3}/6\ , \qquad (4.14b)$$

represent a Mach 10 shock in air ($\gamma = 1.4$), which initially makes a 60 degree angle with the reflecting wall and moves into undisturbed air. Along the left boundary ($x = -0.3$) and the bottom boundary to the left of the wedge ($y = 0$, $-0.3 \le x \le 1/6$) we prescribe Dirichlet boundary conditions according to (4.14); along the top boundary ($y = 1$) values are set to describe the exact motion of an undisturbed Mach 10 shock flow; along the right boundary ($x = 1.9$) all normal derivatives are set to 0; and along the wedge ($y = 0$, $1/6 \le x \le 1.9$) reflecting boundary conditions are used.

This problem was used as a test problem by Woodward and Collela [36] to compare several finite difference schemes on uniform grids for the Euler Equations.

The MacCormack finite difference scheme needs artificial viscosity to 'capture' the shocks of this problem. We used the artificial viscosity developed by Lapidus [26] which is fluid velocity dependent and was used with the MacCormack scheme by Woodward and Collela [36].

The initial mesh used for this problem is 45 X 30, and is shown in Figure 18 (top). We used the magnitude of the density gradient as the mesh movement indicator. From t = 0 to t = 0.01 one error cluster is formed that includes both the Mach 10 shock and the smaller reflected shock region. The resulting mesh at t = 0.01 is shown in Figure 18 (center). This single error cluster is inefficient since it includes different structures of the solution with different propagation velocities. At t = 0.02 the clustering algorithm has recognized two different structures and clusters the error nodes as shown in Figure 18 (bottom). At t = 0.04 three different structures, the Mach 10 shock, the reflected shock, and a Mach stem region, are recognized by the error clustering algorithm as shown in Figure 19 (top). The nodes of the mesh are now able to move with the different velocities of these structures. However, by t = 0.08 it is evident from the mesh of Figure 19 (bottom) that there are not enough nodes to resolve the continuing elongation of the reflected shock region. Mesh refinement is necessary to continue the effective computation of the solution for t > 0.08.

Contour plots of the density at t = 0.08 using the moving mesh and using a uniform stationary mesh with the same number of nodes are compared in Figure 20. The moving mesh reduces the numerical oscillations and provides finer resolution of the shocks and the first contact discontinuity. However, neither calculation was able to resolve

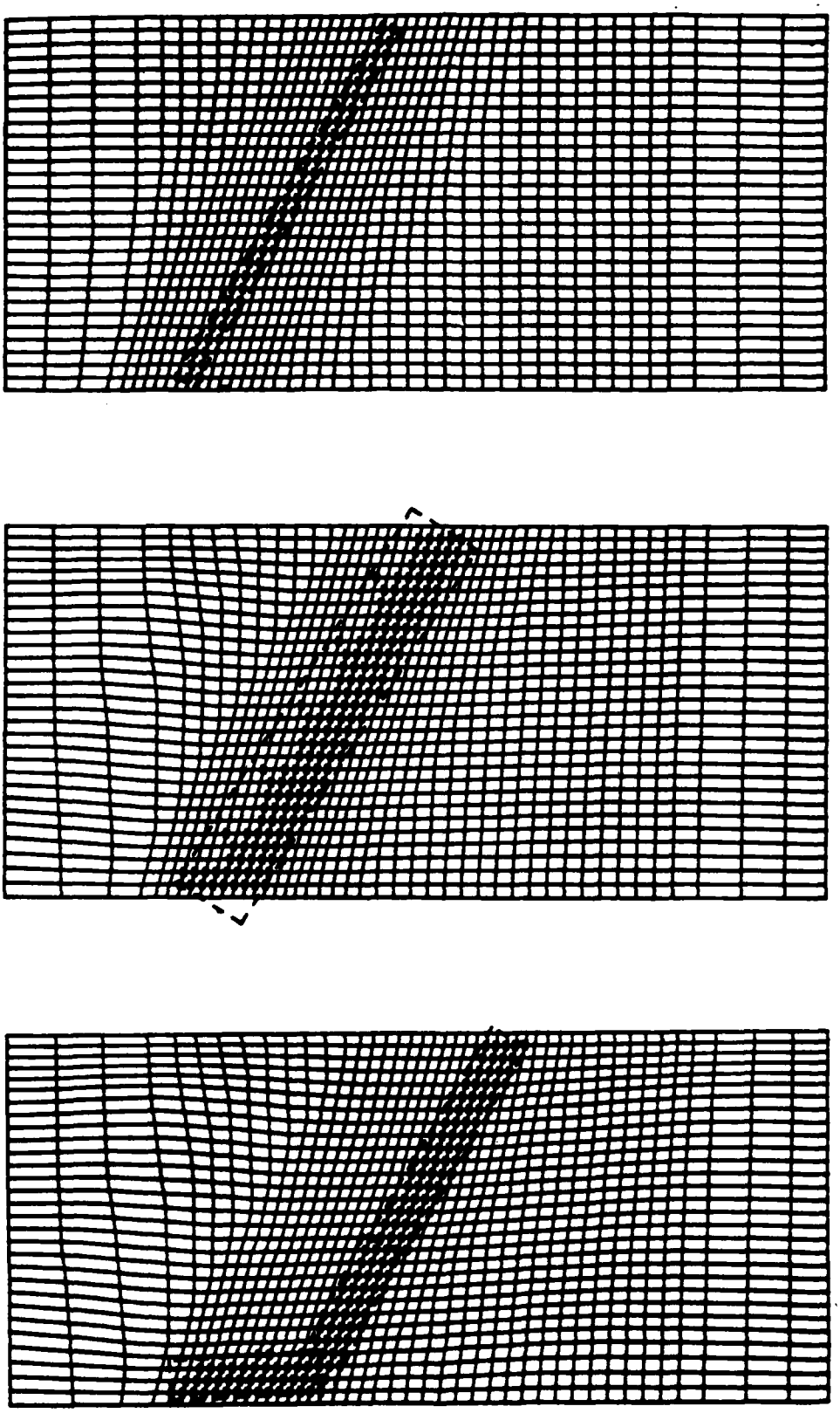


Figure 18. Mesh of Example 4.5 at t = 0.0 (top), at t = 0.01 (center), and at t = 0.02 (bottom)

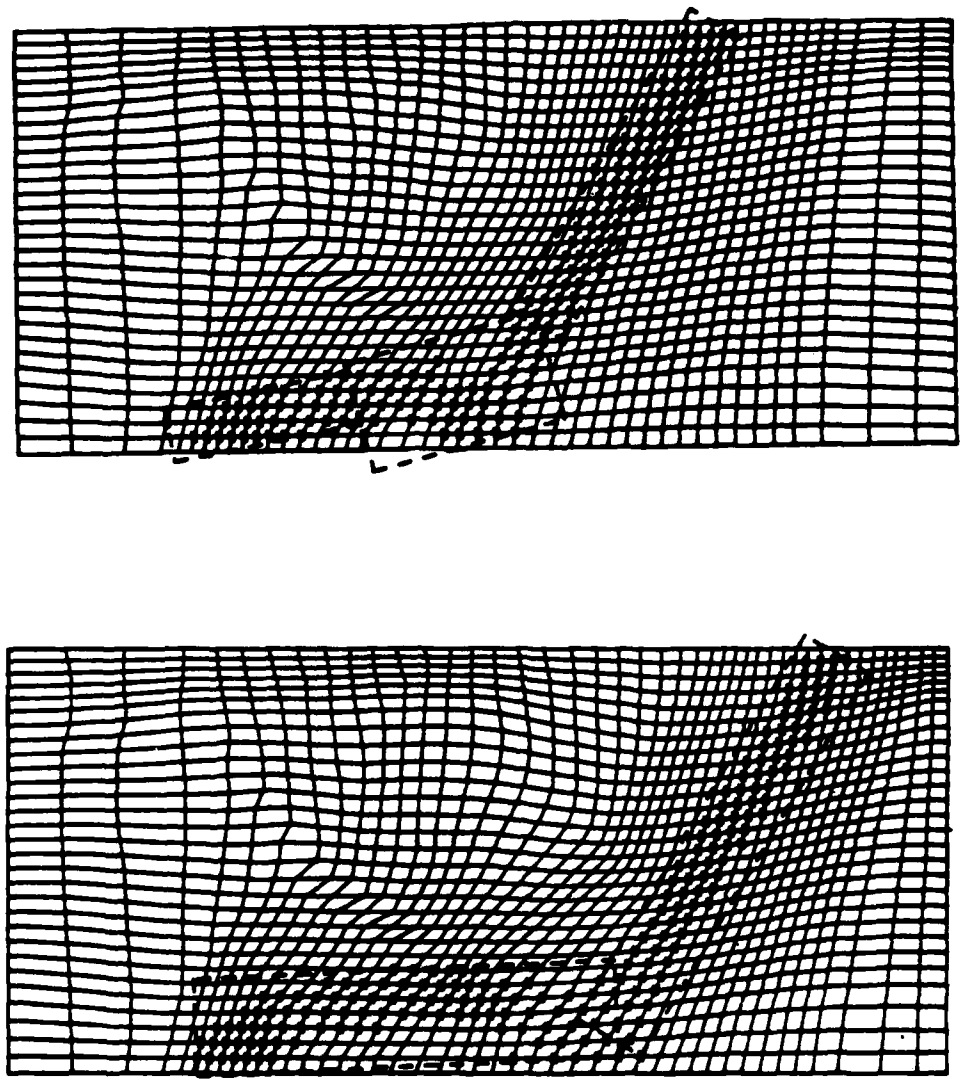


Figure 19. Mesh of Example 4.5 at t = 0.04 (top) and at t = 0.08 (bottom)

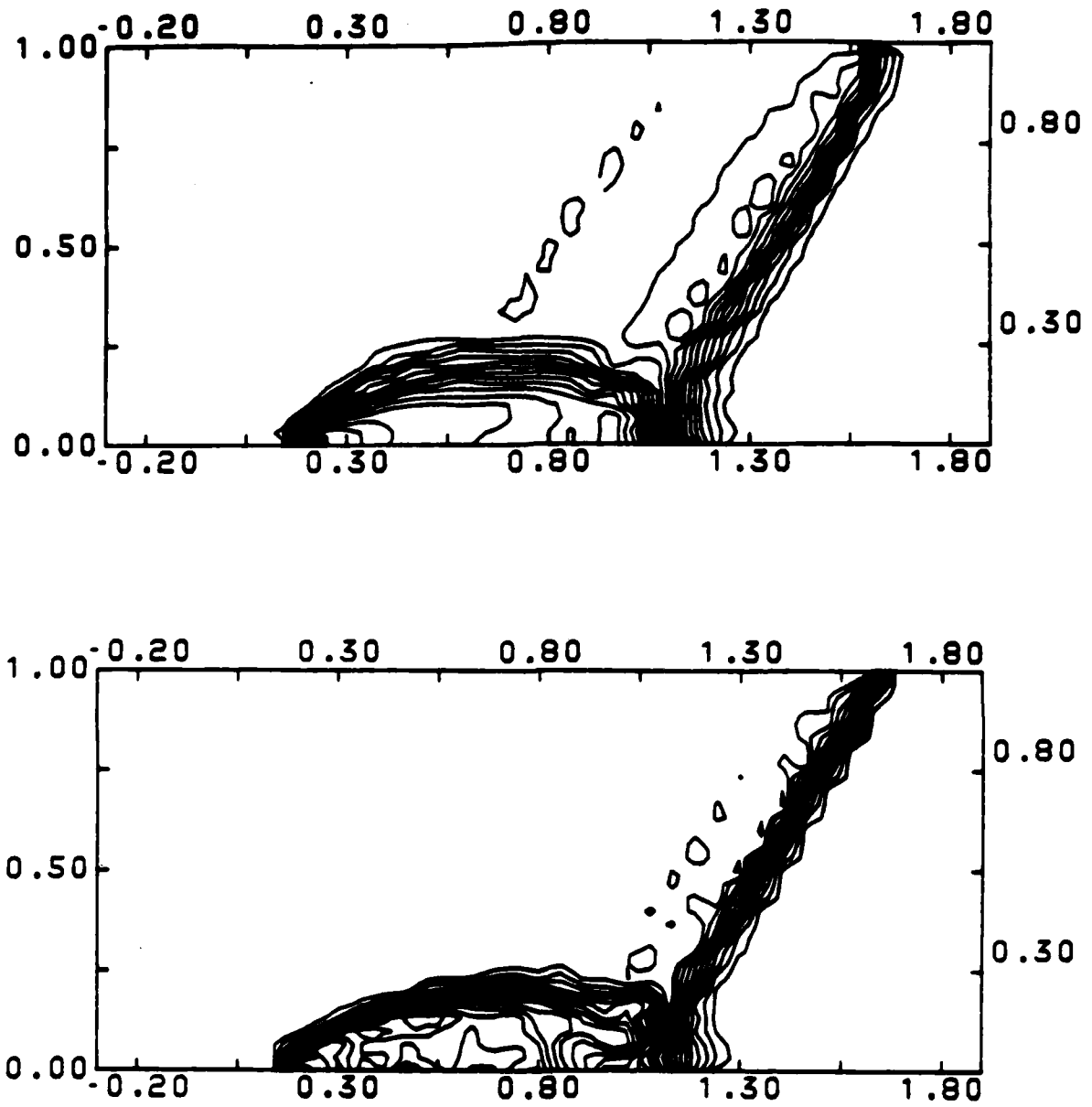


Figure 20. Contour plots of density from the calculated solutions of Example 4.5 at t = 0.08 on a stationary uniform mesh (top) and on the moving mesh as shown in Figures 18 and 19 (bottom).

the fine structures of the second Mach stem and contact discontinuity.

5. CONCLUDING REMARKS

We have described a general two-dimensional mesh moving technique that enables a mesh to follow the propagation of given error indicators. Mesh motion is determined from the movement of clusters of nodes with significantly high error. This procedure was tested on hyperbolic problems having solutions with large gradients.

Even though mesh moving in two dimensions is difficult, we are encouraged by our initial results. The mesh moving algorithm was able to move with the wave and shock fronts of Examples 4.1 and 4.5, was able to control the error rotation of the rotating cone in Example 4.2, and was able to handle the merging and separation of error regions in Example 4.4. The distortion of the mesh in Example 4.3 shows the need for static rezoning when such severe distortions occur. The elongation of the reflected region of Example 4.5 demonstates the need for local mesh refinement in the algorithm.

We are investigating ways to improve the efficiency, reliability, and robustness of the mesh moving algorithm. Possible improvements include (i) not clustering at every time step and letting the mesh move at a constant velocity for several time steps, (ii) efficiently testing for mesh tangling or distortion, and (iii) using a better solver for hyperbolic equations such as the total variation diminishing schemes of Osher [29], van Leer [35], or Engquist [16]. We also hope to show the flexibility of the mesh mover by implementing it with a finite element solver for parabolic problems. Finally, we intend to include local mesh

refinement in the solution algorithm. This combination of mesh moving and refinement should enhance efficiency, accuracy, and robustness.

ACKNOWLEDGEMENT

The authors thank James Hayes, Department of Mathematics, United States Military Academy, for the use of his graphics programs that are included in the system code and were used to plot many of the figures in this paper.

REFERENCES

1. S.ADJERID AND J.E.FLAHERTY, "A Moving Finite Element Method for Time Dependent Partial Differential Equations with Error Estimation and Refinement," preprint, 1984.

2. I.BABUSKA, J.CHANDRA, AND J.E.FLAHERTY, (Eds.) " Adaptive Computational Methods for Partial Differtial Equations," SIAM, Philadelphia,1983.

3. J.B.BELL AND G.R.SHUBIN, *J. Comput. Phys.* 52 (1983), 569.

4. G.BEN-DOR AND I.I.GLASS, *J. Fluid Mech.* 92 (1979),459.

5. M.J.BERGER AND J.OLIGER, *J. Comput. Phys.* 53 (1984),484.

6. M.J.BERGER, "Adaptive Mesh Refinement for Hyperbolic Partial Differential Equations," Ph.D Thesis, Computer Science Department, Stanford University, 1982.

7. M.J.BERGER, in "Adaptive Computational Methods for Partial Differential Equations", (Babuska, I., Chandra, J., and Flaherty, J. E., Eds.), p.237, SIAM, Philadelphia, 1983.

8. M.BIETERMAN AND I.BABUSKA, *Numer. Math.* 40 (1982), 339.

9. M.BIETERMAN AND I.BABUSKA, *Numer. Math.* 40 (1982), 373.

10. J.U.BRACKBILL AND J.S.SALTZMAN, *J. Comput. Phys.* 46 (1982), 342.

11. M.COYLE, J.E.FLAHERTY AND R.LUDWIG, "On the Stability of Mesh

Equidistributing Strategies for Time Dependent Partial Differential Equations," to appear in *J. Comput. Phys.*.

12. S.DAVIS, AND J.E.FLAHERTY, *Siam J. Sci. Stat. Comput.* 3 (1982), 6.

13. D.A.DREW AND J.E.FLAHERTY, "Adaptive Finite Element Methods and the Numerical Solution of Shear Band Problems," to appear in M. Gurtin (Ed.) "Phase Transitions and Material Instabilities in Solids", Academic Press, 1984.

14. J.K.DUKOWICZ, *J. Comput. Phys.* 54 (1984), 411.

15. H.A.DWYER, "Grid Adaption for Problems with Separation, Cell Reynolds Number, Shock-Boundary Layer Interaction, and Accuracy," AIAA Paper No. 83-0449, AIAA Twenty-first Aerospace Sciences Meeting, 1983.

16. B.ENGQUIST AND S.OSHER, *Math. Comp.* 36 (1981), 321.

17. J.E.FLAHERTY AND P.K.MOORE, "An Adaptive Local Refinement Finite Element Method for Parabolic Partial Differential Equations," in "Proc. Conf. Accuracy Estimates and Adaptive Refinements in Finite Element Computations," p.139, Lisbon, 1984.

18. J.E.FLAHERTY, J.M.COYLE, R.LUDWIG, AND S.F.DAVIS, in "Adaptive Computational Methods for Partial Differential Equations," (Babuska, I., Chandra, J., and Flaherty, J. E., Eds.), p.144, SIAM, Philadelphia, 1983.

19. R.J.GELINAS, S.K.DOSS, AND K.MILLER, *J. Comput. Phys.* 40 (1981), 202.

20. D.GANNON, "Self Adaptive Methods for Parabolic Partial Differential Equations," Ph.D. Thesis, Department of Computer Science, University of Illinois at Urbana-Champaign, 1980.

21. D.GOTTLIEB AND S.ORSZAG, "Numerical Analysis of Spectral Methods Theory and Applications," SIAM, Philadelphia, 1977.

22. A.HARTEN AND J.M.HYMAN, *J. Comput. Phys.* 50 (1983), 235.

23. R.HINDMAN, *AIAA J.* 20 (1982), 1359.

24. R.HINDMAN, "A Two-Dimensional Unsteady Euler Equation Solver for Flows in Arbitrarily Shaped Regions using a Modular Concept," Ph.D. Thesis, Iowa State, Ames, Iowa, 1980.

25. J.M.HYMAN, "Adaptive Moving Mesh Methods for Partial Differential Equations," Los Alamos National Laboratory report LA-UR-82-3690, 1982.

26. A.LAPIDUS, *J. Comput. Phys.* 2 (1967), 154.

27. K.MILLER AND R.N.MILLER, *SIAM J. Numer. Anal.* 18 (1981), 1019.

28. K.MILLER, *Siam J. Numer. Anal.* 18 (1981), 1033.

29. S.OSHER AND S.CHAKRAVARTHY, *J. Comput. Phys.* 50 (1983), 447.

30. M.RAI AND D.ANDERSON, "The Use of Adaptive Grids in Conjunction with Shock Capturing Methods," AIAA Paper 81-1012, June 1981.

31. M.RAI AND D.ANDERSON, "Application of Adaptive Grids in Fluid Flow Problems with Asymptotic Solutions," AIAA Paper 81-0114, January 1981.

32. M.RAI AND D.ANDERSON, *J. Comput. Phys.* 43 (1981), 327.

33. J.S.SALTZMAN AND J.BRACKBILL, in "Numerical Grid Generation" (J.F.Thompson, Ed.), p.865, North-Holland, New York, 1982.

34. J.F.THOMPSON, (Ed.), "Numerical Grid Generation", North-Holland, New York, 1982.

35. B.VAN LEER, "Computational Methods for Ideal Compressible Flow," NASA Report 172180, 1983.

36. P.WOOWARD AND P.COLLELA, *J. Comput. Phys.* 54 (1984), 115.

37. O.C.ZIENKIEWICZ, D.W.KELLY, J.GAGO, AND I.BABUSKA, "Hierarchical Finite Element Approaches, Error Estimates and Adaptive Refinement", in "Proc. MAFELAP 1981", April 1981.

38. D.C.ARNEY, Ph.D. Thesis in preparation, Rensselaer Polytechnic Institute, 1985.

LIST OF FIGURES

Figure 12. Comparison of the characteristic path of the peak of the cone and the path of the center of error mass as determined by Equation (2.2) for Example 4.2.

Figure 13. Distorted mesh of Example 4.3 at t = 1.05 showing the need for static rezoning.

Figure 14. Two spatially distinct error clusters formed by the nearest neighbor clustering algorithm for the initial mesh of Example 4.4.

Figure 15. Mesh of Example 4.4 at t = 0.35. The two initial clusters have merged into single cluster centered at the origin.

Figure 16. Mesh of Example 4.4 at t = 0.9. The single cluster separated into two clusters. The two clusters are moving toward the domain boundaries.

Figure 17. Mesh of Example 4.4 at t = 1.4. The two clusters have reached the domain boundaries.

Figure 18. Mesh of Example 4.5 at t = 0.0 (top), at t = 0.01 (center), and at t = 0.02 (bottom)

Figure 19. Mesh of Example 4.5 at t = 0.04 (top) and at t = 0.08 (bottom).

Figure 20. Contour plots of density from the calculated solutions of Example 4.5 at t = 0.08 on a stationary uniform mesh (top) and on the moving mesh as shown in Figures 18 and 19 (bottom).

LIST OF SYMBOLS

ξ := xi

η := eta

τ := tau

ρ := rho

λ := lambda

Σ := upper case sigma, summation symbol

$\mathbf{u}$:= bold face u

$\mathbf{f}$:= bold face f

$\mathbf{g}$:= bold face g

END

FILMED

11-85

DTIC